# Collaborative Exploration by Energy-Constrained Mobile Robots*

Shantanu Das[1], Dariusz Dereniowski[2], and Christina Karousatou[1]

[1] LIF, Aix-Marseille University and CNRS, Marseille, France
[2] Faculty of Electronics, Telecommunications and Informatics, Gdańsk University of Technology, Gdańsk, Poland

**Abstract.** We study the problem of exploration of a tree by mobile agents (robots) that have limited energy. The energy constraint bounds the number of edges that can be traversed by a single agent. Thus we need a team of agents to completely explore the tree and the objective is to minimize the size of this team. The agents start at a single node, the designated root of the tree and the height of the tree is bounded by the energy bound $B$. We provide an exploration algorithm without any knowledge about the tree and we compare our algorithm with the optimal offline algorithm that has complete knowledge of the tree. Our algorithm has a competitive ratio of $O(\log B)$, independent of the number of nodes in the tree. We also show that this is the best possible competitive ratio for exploration of unknown trees.

## 1 Introduction

**Overview:** Graph exploration is a well studied problem in computer science with a wide range of applications from searching the internet to navigation of robots in unknown environments. The objective is to discover an initially unknown graph by visiting all nodes in a systematic manner starting from a given node of the graph. The problem has been well studied for a single agent exploring a graph [16] or a digraph [1] with the aim of minimizing the exploration time or equivalently the number of edges traversed. Others have studied the problem from the perspective of minimizing the memory needed by the agents for exploration [8,13]. When the nodes of the graph do not have identifiers, the agent may need to mark nodes with a pebble to recognize them and thus, another research direction is to minimize the number of pebbles used for exploration [4].

When the exploration is performed by physical robots, one of the major issues is the energy consumed during the exploration, since each robot may have a limited amount of energy for movement. Surprisingly, most previous studies on exploration have not considered this limitation. Betke et al. [5] and later Awerbuch et al. [2] have studied the problem of exploration with an energy constrained agent. Their solution requires a fuelling station at the starting node and

* Partially supported by the ANR projects MACARON (anr-13-js02-0002) and ANCOR (anr-14-CE36-0002-01), and by the Polish National Science Center grant DEC-2011/02/A/ST6/00201.


C. Scheideler (Ed.): SIROCCO 2015, LNCS 9439, pp. 357–369, 2015.
DOI: 10.1007/978-3-319-25258-2_25

the agent periodically returns there to refuel. Between two visits to the starting node, the agent can make at most $B$ edge traversals. Thus the diameter of graphs that can be explored is restricted to $B/2$. When refuelling is not allowed, multiple agents may be needed to explore even graphs of restricted diameter. Given a graph $G$, determining whether a team of $k$ agents, each having an energy constraint of $B$ can explore $G$ is known to be an NP-hard problem, even when the graph $G$ is a tree [12]. When the graph (or the tree) is unknown, there are two possible approaches for online exploration. One approach is to fix the number $k$ of agents and try to bound the amount of energy $B$ required by each agent, as in Dynia et al. [10,11]. In this paper, we take the other approach of fixing the available energy $B$ for each agent and bounding the number of agents used for exploration. Indeed, according to recent trends in robotics [17], it is preferable to use a large number of small robots rather than a few bulky ones and our line of research goes in this direction. In our model, each agent has a limited energy resource without the ability to recharge, thus allowing the agent to traverse at most $B$ edges, and our objective is to limit the total number of such agents used for exploration. We measure the efficiency of the solution in terms of the *competitive ratio* which is defined as the worst case ratio of the cost of the online algorithm for some graph $G$ over the cost of the optimal offline algorithm for the same graph. We restrict ourselves to exploration of trees. The agents start at the designated root of an unknown tree $T$ and they must collectively visit every node of $T$.

> *If the height of $T$ i.e. the longest path between the root and a leaf, is greater than $B$, then $T$ cannot be fully explored, even by an unbounded number of agents.*

On the other hand, if the height of the tree is exactly $B$ then each leaf at depth $B$ must be visited by a separate agent. Once the tree is completely explored and known up to depth $B-1$ then we can send one additional agent to explore each leaf at depth $B$. Thus it is sufficient to consider algorithms for exploring trees of height at most $B-1$. Note that the previous results [2,10] for energy-constrained agents were restricted to exploring trees of height at most $B/2$ (or graphs of diameter at most $B/2$).

**Related Work:** The graph exploration problem has been previously studied with the objective of minimizing the time for exploration. For exploration of undirected graphs by a single agent, the algorithm given by Panaite and Pelc [16] requires $m+O(n)$ time for a graph of $m$ edges and $n$ nodes. For exploration of an unknown tree, exploration in the optimal time of $2(n-1)$ can be achieved by the depth-first search algorithm. Using multiple agents can speedup the exploration and Fraigniaud et al. [12] have presented an algorithm for a team of $k$ mobile agents that explores a tree of height $D$ in $O(D+n/\log k)$ time. They also showed that any algorithm for $k$-agent exploration of a tree has a $(2-1/k)$ overhead over the optimal offline algorithm. While the above results are for small team of agents (where $k \leq \sqrt{n}$), Dereniowski et al. [7] used a large team of agents to

reduce the exploration time to $O(D)$ and their solution also works for general graphs where all nodes are within distance $D$ from the starting node. Ortolf et al. [15] gave bounds on the competitive ratio for multiple agent exploration of grid graphs with obstacles. For general graphs, Megow et al. [14] presented a single-agent exploration algorithm having a constant competitive ratio.

The above results do not consider any energy limitation for the agent. For a single, energy constrained agent, the problem of exploration with refuelling, has been studied for grid graphs [5] and also for general graphs [2]. The optimal time algorithm for exploration with refuelling was given by Duncan et al. [9], who also studied exploration under a different type of constraint where the agent is tied to the starting node with a string of fixed length.

For a team of $k$ agents, the problem of exploring a tree using limited energy resources was investigated by Dynia et al. [10] who presented an algorithm that is 8-competitive in terms of the energy consumed by each agent. This was later improved to a competitive ratio of $(4-2/k)$ by Dynia et al. [11]. Other problems that have been considered for energy constrained agents (that may not start at the same node) include *broadcast* and *convergecast* [3] as well as *data-delivery* from a source node to a target node in the graph [6]. These are mainly offline solutions where the graph and the starting locations are given as input. The algorithm in the present paper can be seen as an online solution to the problem of data-delivery from the root to the leaves or vice versa, for the special case of colocated agents.

**Our Results:** We consider the problem of exploration of an unknown tree by a team of mobile agents initially located at the root of the tree. Each agent is equipped with a battery of size $B$ which bounds the total number of edges the agent can traverse during its lifetime. We assume the height of the tree to be at most $B-1$, and our objective is to find an exploration strategy where every node of the tree is visited by at least one agent, and we wish to minimize the total number of agents used. We study this problem first assuming a global communication model (where agents communicate to each other instantaneously) and provide an algorithm for online exploration, that has a competitive ratio of $O(\log B)$. We then show how to remove the assumption of global communication and achieve the same result in the local communication model, with a constant overhead. Finally we provide a lower bound of $\Omega(\log B)$ on the competitive ratio of any online exploration algorithm for energy-constrained agents, showing that our result is tight. We conclude with some open questions for future research. Due to the space constraint, proofs of some of the lemmas and theorems have been omitted.

## 2 The Model

The environment to be explored is a rooted tree $T$. The root $r_0$ contains an infinite supply of mobile agents, each of which has a limited energy $B$, allowing it to traverse at most $B$ edges during its lifetime. There is a total order among

the agents (i.e. they have distinct identities). The nodes of the tree may be assumed to be anonymous (i.e. we do not require unique identifiers for the nodes of $T$). Each agent has unlimited memory. When two agents are at the same node, they can freely exchange information. However the agents may not write any information on the nodes of the tree. We call this the *local communication* model. In contrast, in a *global communication* model an agent can communicate instantaneously with any other agent irrespective of their location in the tree.

All agents start at the same time, in the same state. At each time unit, any agent can move to an adjacent node or stay at its current node. Each move costs one unit of time and one unit of energy, while computation and communication between agents are instantaneous and do not consume any energy. The agents cannot exchange their energy resources or recharge their batteries.

The height of the tree (i.e. the distance to the furthest leaf from the designated root $r_0$) is at most $B-1$. The size and structure of the tree is initially unknown to the agents. The edges incident at each node are locally ordered with port numbers, allowing the agents to choose edges to visit in a deterministic manner. An exploration strategy for the team of agents is successful if each node of the tree visited by at least one agent. The cost of the exploration strategy is the number of agents which made any non-null moves during the exploration. We denote by $OPT$ the cost of the optimal offline strategy that has complete knowledge of the tree.

For any node $r \in T$ we denote by $T_r$ the subtree of T, rooted at $r$. Further for any node $v \in T_r$, we define the depth of node $v$ as the length of the path from $r$ to $v$. We denote by $T_r^{\delta}$ the subtree rooted at $r$ truncated to depth $\delta$ from $r$. We denote by $|T|$, the number of edges in $T$.

## 3 Exploration with Global Communication

In this section we describe and analyze a recursive algorithm for tree exploration under the global communication model. The algorithm is called *Global Communication Tree Exploration* (GCTE). The main idea of the algorithm is to explore the tree up until a certain depth and afterwards take advantage of the already known part of the tree to continue the exploration. More specifically, this algorithm proceeds by *levels*. Each level of the algorithm is a set of nodes which are located at a certain depth of the tree. The first level consists of the root $r_0$. At each level $i$, agents having energy $b_i$, expand the explored part of the tree further by increasing its depth by $\varepsilon \cdot b_i$ where $\varepsilon$ is a parameter of the algorithm such that $0 < \varepsilon < \frac{1}{4}$. The new frontier of the explored part defines the next level of the algorithm. The algorithm $\texttt{GCTE}_{\varepsilon}$ is then recursively called at each node of the newly created level $i$.

**Definition 1.** *For $i = 1$,* level *$i$ of algorithm $\texttt{GCTE}_{\varepsilon}$ consists of the root node $r_0$; the depth $d_i$ of the level $i$ is $d_1 = 0$, the energy $b_i$ at this level is $b_1 = B$. For $i > 1$,* level *$i$ of $\texttt{GCTE}_{\varepsilon}$ consists of all nodes at depth $d_i = d_{i-1} + \lceil \varepsilon \cdot b_{i-1} \rceil$, and $b_i = B - d_i$.*

For any two nodes $u$ and $v$ at the same level, we would like the exploration of the trees $T_u$ and $T_v$ to proceed independently, using disjoint sets of agents. To this end, we allow some overlap between successive levels of the algorithm. More precisely, at each level $i$, the exploration is extended to the depth of $(\frac{1}{2}+\varepsilon)\cdot b_i$, although the next level still starts at depth $\varepsilon \cdot b_i$ from the current level. This additional extension at each level $i$ allows the algorithm to *look ahead* at the start of the next level $(i+1)$. Thus, at the start of a recursive call to Algorithm $\texttt{GCTE}_\varepsilon$ at a node $r$ at level $i+1$, the subtree $T_r$ has been already partially explored to some depth. We show below (c.f. Lemma 1) that the exploration of this partially explored subtree $T_r$ can be done independently to any other subtree at the same level.

**Definition 2.** *Two partially explored subtrees $T_u$ and $T_v$, rooted at nodes $u$ and $v$ located at the same depth from $r_0$, are said to be* independent *if no single agent can visit nodes in the unexplored part of both subtrees.*

Informally, this independence means that disjoint teams of agents can be used for exploring such subtrees during the algorithm. We now formally describe our algorithm $\texttt{GCTE}_\varepsilon$ .

**Algorithm $\texttt{GCTE}_\varepsilon$ .** An algorithm for tree exploration, $0 < \varepsilon < \frac{1}{4}$

**Input:** The root $r$ of the tree and an integer $b$ that equals the size of the available energy the agents have.

1: Uncover$(r, \lfloor(\frac{1}{2}+\varepsilon)b\rfloor)$
2: Let $r_1, r_2, \ldots$ be nodes at depth $\lceil \varepsilon \cdot b \rceil$ from $r$, such that $T_{r_i}$ has some unexplored edges.
3: For each $r_i$, call Algorithm $\texttt{GCTE}_\varepsilon$ $(r_i, (b - \lceil \varepsilon \cdot b \rceil))$.

Procedure $\texttt{Uncover}(r, \delta)$ with input node $r$ and an integer $\delta$ works as follows. During this procedure, the agents explore the unexplored part of subtree $T_r$ rooted at $r$, using a Depth First Search (DFS) traversal restricted to a depth of $\delta$ from $r$. An agent initially located at the root $r_0$ arrives at the current root $r$, having $b$ units of energy and begins to explore the subtree $T_r^\delta$ performing DFS. First, this agent goes to the next unexplored node in the DFS traversal. At this node, the agent resumes the DFS traversal. Finally, when the agent has $x(\varepsilon) = \frac{1}{2}(\frac{1}{2}-\varepsilon)b$ units of energy left, it interrupts the exploration (it saves the remaining energy for later use, as explained later in the next section). If the point where the agent is supposed to interrupt the DFS traversal is the middle of an edge, then the agent finishes before traversing this edge. Note that in the global communication model, at any point of exploration, each agent possesses the full knowledge of the part of the tree explored to date and the current locations of all agents. Hence, another agent will arrive at $r$ and continue the DFS exploration by visiting the unexplored node that is supposed to be visited next according to the DFS traversal. This procedure ends when all nodes at depth $\delta$ or less have been visited.

**Lemma 1.** *The subtrees that are created in step 2 of* $\texttt{GCTE}_\varepsilon$ *are pairwise independent. Moreover, for any such subtree* $T_r$ *rooted at a node* $r$ *any agent that reaches the unexplored part of* $T_r$ *cannot return to node* $r$*.*

**Theorem 1.** *For any* $\varepsilon$, $0 < \varepsilon < \frac{1}{4}$, *Algorithm* $\texttt{GCTE}_\varepsilon$ *called for* $r_0$ *and* $B$ *correctly explores the tree.*

*Proof.* To prove the correctness of $\texttt{GCTE}_\varepsilon$, we first show that procedure $\texttt{Uncover}(r, \delta)$ with $\delta = \lfloor (\frac{1}{2} + \varepsilon)b \rfloor$ correctly explores the subtree rooted at node $r$ up to depth $\delta$. Note that by a simple induction on the distance of $r$ from $r_0$, any agent that arrives at node $r$, to execute $\texttt{Uncover}(r, \delta)$, has exactly $b$ units of energy. Further any such agent $A_j$ has complete knowledge of the part of subtree $T_r$ already explored by previous agents and thus agent $A_j$ knows the path from $r$ to the next unexplored node $v$ in the DFS traversal of $T_r^\delta$. This node $v$ must be at distance at most $\delta$ from $r$. According to the algorithm, the agent uses

$$l := b - \lceil x(\varepsilon) \rceil = \lfloor b - x(\varepsilon) \rfloor = \lfloor \delta + x(\varepsilon) \rfloor$$

units of energy during the DFS traversal. Since $l \geq \delta$ the agent does succeed in reaching the node $v$. Hence, each agent used in `Uncover` visits at least one previously unexplored node in $T_r^\delta$. This implies that eventually all nodes within depth $\delta$ in $T_r$ are visited during the DFS exploration. This proves the correctness of procedure `Uncover`.

In order to complete the proof of the correctness of $\texttt{GCTE}_\varepsilon$, we note that the algorithm makes progress at each level $i$, that is, level $i+1$ is at strictly greater depth than level $i$. Indeed, this follows from $\varepsilon b_i > 0$ for $\varepsilon > 0$, which gives $\lceil \varepsilon b_i \rceil \geq 1$. □

**Lemma 2.** *The number of levels in Algorithm* $\texttt{GCTE}_\varepsilon$ *is at most* $\log_{(\frac{1}{1-\varepsilon})} B$*.*

Before proceeding to calculating the cost of Algorithm $\texttt{GCTE}_\varepsilon$, let us make the following useful remark. During the procedure `Uncover`, each participating agent uses at most $\delta$ energy to reach the starting node for its DFS exploration and uses at least $b - \delta - \lceil x(\varepsilon) \rceil = \lfloor \frac{1}{2}(\frac{1}{2} - \varepsilon)b \rfloor$ units of energy to contribute to the DFS exploration of unexplored nodes.

**Lemma 3.** *Procedure* `Uncover`$(r,\delta)$ *for* $r = r_0$ *and* $\delta = (1/2+\varepsilon)B$ *uses* $SOL_r \leq \frac{4}{(\frac{1}{2}-\varepsilon)} \cdot OPT$ *agents.*

**Theorem 2.** *Algorithm* $\texttt{GCTE}_\varepsilon$ *has a competitive ratio of* $\frac{4}{(\frac{1}{2}-\varepsilon)} \cdot \log_{(\frac{1}{1-\varepsilon})} B$*.*

*Proof.* Consider a call to $\texttt{GCTE}_\varepsilon(r, b_i)$ at some level $i > 1$, where $r$ is at depth $d_i > 0$ from the global root. Let $SOL_r$ denote the number of agents used by the algorithm to explore edges of the subtree $T_r$ during level $i$. A DFS exploration walk of $T_r$ that starts and ends at $r$ has length $2 \cdot |T_r|$. As explained before

each of the $SOL_r$ agents (except the last one) use at least $\frac{1}{2}(\frac{1}{2}-\varepsilon)b_i$ of their available energy to contribute to the DFS exploration. The last agent may have some available energy after visiting the last unexplored edge in $T_r$ but it does not have enough energy to return to node $r$ (by Lemma 1). Thus if we assume that the last agent attempts to reach the root $r$ with its remaining energy, we can say that the path traversed in total by the agents is at most $2 \cdot |T_r|$. Thus,

$$\frac{1}{2}(\frac{1}{2}-\varepsilon)b_i \cdot SOL_r \leq 2 \cdot |T_r| \implies SOL_r \leq \frac{4}{b_i(\frac{1}{2}-\varepsilon)}|T_r|$$

Furthermore, due to Lemma 1, we know the subtrees at the same level are independent so we can sum up over all subtrees at level $i$:

$$\sum_{r \in r_1, r_2, \ldots} SOL_r \leq \frac{4}{b_i(\frac{1}{2}-\varepsilon)} \sum_{r \in r_1, r_2, \ldots} |T_r|$$

$$SOL(i) \leq \frac{4}{b_i(\frac{1}{2}-\varepsilon)}|T \setminus T^{d_i}|$$

where $SOL(i)$ denotes the number of agents used by the algorithm at level $i$. The optimal algorithm uses $OPT$ agents to explore the tree. Any agent that reaches to depth $d_i$ of $T$ has $b_i$ units of energy remaining. Thus, each agent can traverse at most $b_i$ edges below this depth. Hence

$$b_i \cdot OPT \geq |T \setminus T^{d_i}|$$

Combining the above two equations, we have

$$SOL(i) \leq \frac{4}{\frac{1}{2}-\varepsilon}OPT$$

The above bound holds for any level $i > 1$. Moreover, due to Lemma 3, we have exactly the same bound for level $i = 1$ of the algorithm. Since there are at most $\log_{(\frac{1}{1-\varepsilon})} B$ levels in the algorithm (due to Lemma 2), we obtain the total cost $SOL$ of the algorithm,

$$SOL \leq \frac{4}{\frac{1}{2}-\varepsilon} \cdot \log_{(\frac{1}{1-\varepsilon})} B \cdot OPT$$

□

Note that on the termination of algorithm $\texttt{GCTE}_\varepsilon$ , each agent that participated in the exploration at level $i$ has at least $x_i(\varepsilon) = \frac{1}{2}(\frac{1}{2}-\varepsilon) \cdot b_i$ units of unused energy. This remaining energy would be used by the algorithm presented in the next section.

## 4 Exploration with Local Communication

This section is devoted to adaptation of $\texttt{GCTE}_\varepsilon$ for the model with local communication between agents. This is done in two steps. In the first step we introduce an intermediate stage between two models of global and local communication. We call this a *semi-local communication model* and we define it as follows: two agents performing the DFS exploration in Step 1 of an instance of $\texttt{GCTE}_\varepsilon$ can communicate only locally, that is, they can communicate only when present at the same node; on the other hand, the algorithm may call for a new agent that is placed at the root $r$ of a subtree explored by an instance of $\texttt{GCTE}_\varepsilon$. Note that, when an instance of $\texttt{GCTE}_\varepsilon$ calls for a new agent to arrive at the input node $r$, this agent is initially present at the 'global' root of the entire tree and needs to traverse the path from the global root to $r$. Thus, in our semi-local communication model this mechanism of calling for agents uses the global communication model. In Section 4.1 we adopt $\texttt{GCTE}_\varepsilon$ so that it operates in the semi-local communication model and we calculate the cost of this modification in terms of the number of agents used. In particular, we prove that with respect to the original algorithm, the total number of agents increases by a constant factor (depending only on $\varepsilon$). Then, in Section 4.2, we add to our algorithm a mechanism for calling for new agents at local roots so that this part is also done via local communication.

### 4.1 Semi-local Communication Model

We start this section by providing some intuition. We consider an arbitrary execution of $\texttt{GCTE}_\varepsilon(r, b)$ for an input node $r$ and energy level $b$. Recall Step 1 of $\texttt{GCTE}_\varepsilon$, where the agents, one by one, perform the DFS traversal up to a certain depth of the subtree $T_r$. Suppose that the agents that perform this traversal are $A_1, \ldots, A_k$ and that they are ordered according to the precedence of their movements, i.e., $A_i$ traverses its path prior to $A_{i+1}$ for each $i \in \{1, \ldots, k-1\}$. For each agent $A_i$ we will add a constant number of $c(\varepsilon)$ additional agents denoted $A_i^1, \ldots, A_i^{c(\epsilon)}$, where

$$c(\varepsilon) = 2 \cdot \left\lceil \frac{1/2 + \varepsilon}{1/2 - \varepsilon} \right\rceil, \quad 0 < \varepsilon < 1/4. \tag{1}$$

To simplify some statements we sometimes write $A_i^0$ in place of $A_i$. The agents $A_i, A_i^1, \ldots, A_i^{c(\varepsilon)}$ are called the *$i$-th team* for each $i \in \{1, \ldots, k\}$. For the purposes of the analysis we introduce some additional notation that allows us to describe the behavior of agents during this DFS traversal in more details. We denote by brevity

$$x(\varepsilon) = \frac{1}{2}\left(\frac{1}{2} - \varepsilon\right) b. \tag{2}$$

We also say that an agent *heads towards* a node $v$ if in each of the following consecutive time units the agent makes a move that gets it closer to $v$ until either $v$ is reached or the agent runs out of energy. It is said that the $i$-th team

is *successful* if: (i) the agent $A_i$ visited a superset of nodes with respect to its original behavior in Step 1 of $\texttt{GCTE}_\varepsilon$, and (ii) the agent $A_i^{c(\varepsilon)}$ reaches the root $r$ and possesses the information about all moves performed by agents $A_1, \ldots, A_i$.

We now describe the modification of the DFS traversal from Step 1 of $\texttt{GCTE}_\varepsilon$ by describing how $A_i$ and $A_i^1, \ldots, A_i^{c(\varepsilon)}$ operate for each $i \in \{1, \ldots, k\}$.

**Behavior of $A_i$.** Recall that in Step 1 of $\texttt{GCTE}_\varepsilon$, each agent $A_i$, $i \in \{1, \ldots, k\}$, finishes its part of DFS traversal having at least $x(\varepsilon)$ energy left. We now use this energy as follows: the agent heads towards the root $r$ in the next $\lceil x(\varepsilon) \rceil$ time units.

**Behavior of $A_i^j$'s.** For each $i \in \{1, \ldots, k\}$ and $j \in \{1, \ldots, c(\varepsilon)\}$, the agent $A_i^j$ follows the movements of $A_i$ up to the depth

$$d_j(\varepsilon) = \lfloor j \cdot x(\varepsilon) \rfloor$$

until the completion of the movement of $A_i$. More precisely, the agent $A_i^j$ mimics each move of $A_i$ from node $u$ to node $v$ if both $u$ and $v$ are within depth (from $r$) at most $d_j(\varepsilon)$. If, on the other hand, either $u$ or $v$ is at depth greater than $d_j(\varepsilon)$, then $A_i^j$ stays idle in this given time unit. Finally, the agent $A_i^j$ heads towards the root $r$; we will describe below in which time unit this action is triggered.

**Order of Movements.** Having described the movements of $A_i$ and $A_i^j$ for each $i \in \{1, \ldots, k\}$ and $j \in \{1, \ldots, c(\varepsilon)\}$, we specify the order of their actions. The agent $A_1$ starts its movement once all agents of the 1-st team are at $r$. For each $i \in \{2, \ldots, k\}$, the agent $A_i$ starts its movement once $A_{i-1}^{c(\varepsilon)}$ completed its movement by arriving at $r$ and once all agents of the $i$-th team are at $r$. (We will argue later that $A_{i-1}^{c(\varepsilon)}$ indeed returns to the root $r$.) In other words, once $A_{i-1}^{c(\varepsilon)}$ completes its movement, all agents of the $i$-th team are called to appear at $r$. For each $j \in \{1, \ldots, c(\varepsilon)\}$, we only need to describe how they operate once $A_i$ runs out of energy, as their preceding movements are specified above. The agent $A_i^j$ heads towards the root $r$ in time unit in which he occupies the same node as $A_i^{j-1}$ and the latter agent is heading towards $r$. (Thus, it may happen that for a number of time units both agents will head towards $r$ together.)

In the following we prove that the above actions of agents are valid under the assumption that they have to communicate locally. Considering the order of movements of agents it suffices to argue that each team is successful. We refer to all movements of the agents $A_i^j$, $i \in \{1, \ldots, k\}$, $j \in \{1, \ldots, c(\varepsilon)\}$, as the *extended DFS traversal* of $T$.

**Lemma 4.** *For each $i \in \{1, \ldots, k\}$, the $i$-th team is successful.*

As a consequence of the above, we obtain the following.

**Lemma 5.** *The extended DFS traversal correctly explores $T_r$ to a depth of $(\frac{1}{2} + \varepsilon)B$ using $k(c(\varepsilon)+1)$ agents that communicate locally, where $k$ is the number of agents used in the DFS traversal performed in Step 1 of Algorithm $\texttt{GCTE}_\varepsilon$.*

*Proof.* The fact that the $k$ teams, each of size $c(\varepsilon) + 1$, explore the tree to the required depth follows directly from Lemma 4. □

### 4.2 Local Communication between Levels

We start this section with an informal description, also pointing out the obstacles we need to overcome. The mechanism of communication between two consecutive levels will be handled by special agents that we call *managing agents* (see below for a formal definition). A managing agent arrives at a root $r$ for which a call to $\texttt{GCTE}_\varepsilon$ is performed. This agent is not used for the extended DFS traversal of $T_r$ but will play a crucial role while conducting recursive calls for descendants $r_1, r_2, \ldots$. More precisely, this agent will keep track of which subtrees have been already explored and for which one the recursive call is 'in progress'. By a recursive call, made say for $r_i$, being in progress we mean that the exploration of $T_{r_i}$ is in progress. Thus, until the exploration of that subtree is completed, the managing agent for $T_r$ is responsible for redirecting all agents arriving at $r$ to this subtree $T_{r_i}$. Once the exploration of $T_{r_i}$ is completed, the managing agent for $T_{r_i}$ will report this fact to the managing agent for $T_r$ and the latter one may initiate the process of exploration of the next subtree $T_{r_{i+1}}$. Once all subtrees $T_{r_1}, T_{r_2}, \ldots$ are explored the managing agent for $T_r$ returns 'one level up' to report this event to appropriate managing agent.

Observe that the above scheme should be performed in such a way that each subtree $T_r$ 'receives' just enough agents needed for its exploration and not more. This includes one managing agent for the subtree itself, the agents performing the extended DFS traversal of $T_r$ and the agents needed for recursive calls, if any. This is regulated by introducing the agents slowly at the global root so that, within predefined time intervals new agents appear at the global root and are directed gradually by managing agents precisely to the subtree for which the current extended DFS traversal is performed. The time intervals are set up in such a way that if an exploration of a particular subtree is completed then this information has enough time to be carried by the managing agent to the one residing one level up. In this way the flow of agents to a particular subtree is stopped and redirected to the next one supplying the exact amount of agents needed for each of the subtrees. Intuitively, the measurement of time is used indirectly as a communication tool: if a managing agent does not receive for a given amount of time a signal that a recursive call to a subtree is completed, then this means that the exploration of that subtree is not completed and more agents are needed to finish it — hence another agent will be sent to that subtree.

Now we give a detailed description of the modifications to the exploration strategy described in Section 4.1 so that it is valid for agents communicating locally. At the beginning of exploration (i.e., when $\texttt{GCTE}_\varepsilon$ is called for a tree $T$), one distinguished agent is selected to be constantly present at the root $r_0$ of the entire tree $T$. This agent is called the *managing agent for* $T$. Similarly, whenever a recursive call of $\texttt{GCTE}_\varepsilon$ is made for any input node $r$, the first agent that arrives at $r$ is the managing agent for $T_r$ and it stays at $r$ until the entire subtree $T_r$ is explored.

**Extension of Step 1 of $\texttt{GCTE}_\varepsilon$.** Once all $c(\varepsilon) + 1$ members of the $i$-th team are present at the root $r$ of a subtree for which the extended DFS traversal is performed, the $i$-th team operates exactly as described in Section 4.1. Recall

that the $i$-th team finishes its work with one of its agents being at the root. The beginning of the operation of the $(i+1)$-th team is postponed until exactly $c(\varepsilon)+1$ new agents, each with energy $b$, appear at $r$. Then, the $(i+1)$-th team resumes the extended DFS traversal. We note that the agents forming each team will arrive at $r$ directly from the global root of the tree and this will become clear after description of the extension of Step 3 of $\texttt{GCTE}_\varepsilon$.

**Extension of Step 3 of $\texttt{GCTE}_\varepsilon$.** For this part we need to describe how a recursive call is performed by an instance of $\texttt{GCTE}_\varepsilon$. This includes two actions: initiating the call and receiving information that a recursive call is completed, i.e., that the exploration of the subtree for which the call was conducted is finished. Suppose that an instance of $\texttt{GCTE}_\varepsilon$ with input $r$ and $b$ performs a call for a subtree rooted at a node $r_i$. Recall that the managing agent for $T_r$, denoted by $A(r)$ is present at $r$ during exploration of $T_r$. First, $A(r)$ waits until a new agent, denoted by $A(r_i)$, appears at $r$ and after this event this agent is sent to $r_i$ and it becomes the managing agent for $T_{r_i}$. Then, the algorithm sends each agent arriving at $r$ to the node $r_i$ until the agent $A(r_i)$ returns to $r$. This completes the recursive call for $r_i$ and $A(r_i)$ stays idle at $r$ indefinitely (and will not play any role in the remaining part of the exploration). Then, the next recursive call, if any, that needs to be done is performed. The information about the current status of each recursive call made by the instance of $\texttt{GCTE}_\varepsilon(r,b)$, is maintained by $A(r)$, the managing agent for $T_r$, and once all recursive calls are completed this managing agent returns to the node that is the ancestor of $r$ from which the instance of $\texttt{GCTE}_\varepsilon(r,b)$ was called.

**Distribution of agents at the global root.** Note that the above description defines the operation of agents for each instance of $\texttt{GCTE}_\varepsilon$ except for the managing agent at the global root $r_0$ for the first call to $\texttt{GCTE}_\varepsilon$. The managing agent at the global root has all agents at its disposal from the first step and does not need to wait for the arrival of an agent. Therefore we introduce an artificial delay denoted by $d(\varepsilon)$ as defined below. The $d(\varepsilon)$ is an integer and it will be understood that the agents will appear at the global root $r$ in time intervals of $d(\varepsilon)$. This time interval is defined as

$$d(\varepsilon) = (c(\varepsilon)+2)B. \tag{3}$$

The exploration strategy modified as above is called $\texttt{LCTE}_\varepsilon$ (*Local Communication Tree Exploration*). We now prove that $\texttt{LCTE}_\varepsilon$ works correctly in the local communication model.

**Lemma 6.** *For $0 < \varepsilon < 1/4$, Algorithm $\texttt{LCTE}_\varepsilon$ correctly explores any tree $T$ using local communication between agents.*

**Theorem 3.** *Algorithm $\texttt{LCTE}_\varepsilon$ explores $T$ using at most $O(\log B)\cdot OPT$ agents.*

## 5 Competitive Ratio of Online Exploration

We now show a lower bound on the competitive ratio of any online exploration algorithm in the local communication model. The following result implies that the competitive ratio of algorithm $\texttt{LCTE}_\varepsilon$ is asymptotically optimal.

**Theorem 4.** *Any online exploration algorithm for exploring a tree of depth* $D = B - 1$ *has a worst case competitive ratio of at least* $\Omega(\log B)$.

*Proof.* We consider the family of trees which consist of a line of length $D - 1$ connected to the center of a star with $p$ leaves. Thus all the $p$ leaves of the tree are at distance $D = B - 1$ from the root and there is only one node at distance $D-1$. An offline algorithm would use exactly $p$ agents for exploring this tree. An online algorithm for exploring this tree can be of two types: We say an algorithm is type-1 if during the algorithm there is no transfer of information from the node at depth $D - 1$ to the root; All other algorithms are of type-2. First notice that if an algorithm of type-1, uses $k$ agents for exploration then $k$ is independent of $p$, since $p$ remains unknown to the root. Thus, by taking $p > k$, we can make the algorithm fail. So we need to consider only type-2 algorithms where information from the node at depth $D - 1$ is transferred to the root. Any agent visiting this node has at most $B - (D - 1) = 2$ units of energy remaining, so it can return back to depth $D - 3 = B - 4$. Similarly, any agent visiting the node at depth $B - 4$ can return back to depth $B - 8$, and so on. Thus, at least $\Omega(\log B)$ agents are needed to carry the information from the node at depth $D - 1$ back to the root. So any type-2 algorithm would use at least $\Omega(\log B)$ agents. By taking $p = 1$, we get a competitive ratio of $\Omega(\log B)$ for any such algorithm. □

## 6 Conclusions

We studied the problem of exploring a tree with a team of agents, each of which can traverse at most $B$ edges. We gave matching lower and upper bound of $\Theta(\log B)$ on the competitive ratio of the cost of tree exploration. Unlike previous algorithms for energy constrained agents, the agents in our algorithm do not necessarily return to the root after exploration. This fact allows us to explore trees of larger depth. However there is still a transfer of information from the leaves to the root. Thus the algorithm can be used e.g. to collect information from the leaves of a tree, or to search for a resource and bring it back to the root. Note that the lower bound of $\Omega(\log B)$ on the competitive ratio holds only in the local communication model. An interesting question is whether more efficient algorithms are possible for tree exploration in the global communication model. Another open question is the cost of exploring general graphs or other specific classes of graphs.

## References

1. Albers, S., Henzinger, M.R.: Exploring Unknown Environments. SIAM Journal on Computing 29(4), 1164–1188 (2000)
2. Awerbuch, B., Betke, M., Singh, M.: Piecemeal graph learning by a mobile robot. Information and Computation 152, 155–172 (1999)
3. Anaya, J., Chalopin, J., Czyzowicz, J., Labourel, A., Pelc, A., Vaxés, Y.: Convergecast and Broadcast by Power-Aware Mobile Agents. Algorithmica, 1–39 (2014)

4. Bender, M., Fernandez, A., Ron, D., Sahai, A., Vadhan, S.: The power of a pebble: Exploring and mapping directed graphs. In: Proc. 30th ACM Symp. on Theory of Computing (STOC), pp. 269–287 (1998)
5. Betke, M., Rivest, R.L., Singh, M.: Piecemeal learning of an unknown environment. Machine Learning 18(23), 231–254 (1995)
6. Chalopin, J., Das, S., Mihalák, M., Penna, P., Widmayer, P.: Data delivery by energy-constrained mobile agents. In: Flocchini, P., Gao, J., Kranakis, E., der Heide, F.M.a. (eds.) ALGOSENSORS 2013. LNCS, vol. 8243, pp. 111–122. Springer, Heidelberg (2014)
7. Dereniowski, D., Disser, Y., Kosowski, A., Pająk, D., Uznański, P.: Fast collaborative graph exploration. Information and Computation 243, 37–49 (2015)
8. Diks, K., Fraigniaud, P., Kranakis, E., Pelc, A.: Tree exploration with little memory. Journal of Algorithms 51, 38–63 (2004)
9. Duncan, C.A., Kobourov, S.G., Anil Kumar, V.S.: Optimal constrained graph exploration. In: Proc. 12th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 807–814 (2001)
10. Dynia, M., Korzeniowski, M., Schindelhauer, C.: Power-aware collective tree exploration. In: Grass, W., Sick, B., Waldschmidt, K. (eds.) ARCS 2006. LNCS, vol. 3894, pp. 341–351. Springer, Heidelberg (2006)
11. Dynia, M., Łopuszański, J., Schindelhauer, C.: Why robots need maps. In: Prencipe, G., Zaks, S. (eds.) SIROCCO 2007. LNCS, vol. 4474, pp. 41–50. Springer, Heidelberg (2007)
12. Fraigniaud, P., Gasieniec, L., Kowalski, D., Pelc, A.: Collective tree exploration. Networks 48(3), 166–177 (2006)
13. Fraigniaud, P., Ilcinkas, D., Peer, G., Pelc, A., Peleg, D.: Graph exploration by a finite automaton. Theoretical Computer Science 345(2-3), 331–344 (2005)
14. Megow, N., Mehlhorn, K., Schweitzer, P.: Online graph exploration: new results on old and new algorithms. In: Aceto, L., Henzinger, M., Sgall, J. (eds.) ICALP 2011, Part II. LNCS, vol. 6756, pp. 478–489. Springer, Heidelberg (2011)
15. Ortolf, C., Schindelhauer, C.: Online multi-robot exploration of grid graphs with rectangular obstacles. In: Proc. 24th ACM Symp. on Parallelism in Algorithms and Architectures (SPAA), pp. 27–36 (2012)
16. Panaite, P., Pelc, A.: Exploring unknown undirected graphs. Journal of Algorithms 33, 281–295 (1999)
17. Rutishauser, S., Correll, N., Martinoli, A.: Collaborative Coverage using a Swarm of Networked Miniature Robots. Robotics and Autonomous Systems 57(5), 517–525 (2009)