Energy-Optimal Broadcast in a Tree with Mobile Agents

Jerzy Czyzowicz[2]⋆, Krzysztof Diks[1]⋆⋆, Jean Moussi[2], and Wojciech Rytter[1]⋆⋆⋆

[1] Département d'informatique, Université du Québec en Outaouais, Gatineau, Québec, Canada
[2] Faculty of Mathematics, Informatics and Mechanics, University of Warsaw, Warsaw, Poland
[diks,rytter]@mimuw.edu.pl [jurek,Jean.Moussi]@uqo.ca

Abstract. A set of k mobile agents is deployed at the root r of a weighted, n-node tree T. The weight of each tree edge represents the distance between the corresponding nodes along the edge. One node of the tree, the source s, possesses a piece of information which has to be communicated (broadcasted) to all other nodes using mobile agents. An agent visiting a node, which already possesses the information, automatically acquires it and communicates it to all nodes subsequently visited by this agent. The process finishes when the information is transferred to all nodes of the tree.
The agents spend energy proportionally to the distance traversed. The problem considered in this paper consists in finding the minimal total energy, used by all agents, needed to complete the broadcasting. We give an $O(n \log n)$ time algorithm solving the problem. If the number of agents is sufficiently large (at least equal to the number of leaves of T), then our approach results in an $O(n)$ time algorithm. When the source of information s is initially at the root r, our algorithm solves the problem of searching the tree (exploring it) by a set of agents using minimal energy. It is known that, even if the tree is a line, the broadcasting problem and the search problem are NP-complete when the agents may be initially placed at possibly many distinct arbitrary positions.

Keywords: mobile agents, tree, data delivery, broadcast, search

1 Introduction

A packet of information available at the source node of a network must be disseminated to all other nodes. The task needs to be performed by a collection of mobile agents. Given a network, what is the minimal amount of energy needed for the information to be delivered to all nodes. If the agents are initially distributed along the network, this problem turns out to be NP-complete, even if the original network is a line (see [13]). In this paper we solve the problem for tree networks, when all agents start at the same node.

Similar problems, studied in operations research (particularly in vehicle routing), sometimes assume limited capacities of robots and quantities of product to be transported. In our case, the amount of product to be carried by a robot is irrelevant. Consequently, similarly to [13], we categorize our problem as *data delivery* or, more exactly as *broadcasting* using mobile agents.

Moreover, in the special case when the data source and the starting position of the robots are the same, our solution produces the optimal search schedule - the movement of the collection of agents searching all nodes of the tree using minimal total energy. It is a folklore knowledge that the same problem when the time of the schedule is to be optimized (i.e. the time of arrival to its destination of the last robot), the problem is NP-complete.

⋆ Supported by NSERC grant of Canada
⋆⋆ Supported by the grant NCN2014/13/B/ST6/00770 of the Polish Science Center.
⋆ ⋆ ⋆ Supported by the grant NCN2014/13/B/ST6/00770 of the Polish Science Center.

1.1 Preliminaries and the problem statement

A set of k mobile agents is placed at the root r of an edge-weighted tree T, where edge weight represents distance between edge endpoints. One node of the tree, that we call *source node*, initially possesses a packet of information, which eventually needs to reach every other node of the tree. The information is transported by mobile agents, which use energy proportionally to the distance travelled. We conservatively assume, that if an agent previously acquired the source information packet, then if such agent is visiting a node, it implicitly leaves a copy of the packet at that node. Any agent later visiting such node automatically copies the packet to its memory and it may then distribute it to other, subsequently visited nodes. We consider the following problem:

Data broadcasting:
Let T be an edge-weighted tree with two distinguished nodes s, r. The node s is a source node and r is the initial location of k mobile agents. The source node possesses a packet of information, which needs to be transported to all other nodes by mobile agents.
What is the minimal amount of energy, denoted by $MinCost(T, k)$, that the agents need for their travel so that the source packet is successfully distributed?

By a (global) schedule we mean a set of functions $f_1, f_2, \dots, f_k$, such that $f_i(t)$ is a position at time t of agent i in the tree. The agent i knows the data packet at time t^*, if $f_i(t) = s$ for some $t \leq t^*$, or $f_i(t_1) = v$, $f_j(t_2) = v$ and $t_2 \leq t_1 \leq t^*$ for some agent j, which knew the packet at time t_2.

We call a schedule *optimal* if it results in the smallest possible usage of energy by mobile agents.

1.2 Our results

We present almost linear-time, greedy algorithms solving data broadcasting. Our problem can be solved in $O(n \log n)$ time, independently of the number k of available agents. This complexity is reduced to $O(n)$ time in case of unlimited number of agents, or when the number of agents is at least equal to the number of leaves of T.

In the special case when the root, from which all agents start, is also the source node, our approach solves the search problem, when the collection of agents need to search the tree optimally, i.e. using the smallest total energy. Surprisingly, according to our knowledge, this natural setting of the search/exploration problem has not been studied before.

The missing proofs will appear in the full version of the paper.

1.3 Related work

Recent development of the network technology fuelled the research in mobile agents computing. Several applications involve physical mobile devices, software agents, migrating in a network, or living beings: humans (e.g. soldiers or disaster relief personnel) or animals (e.g. ants). Most important questions for mobile agents concern environment search or exploration (cf. [3, 9, 15–17]). Some questions involving mobile agents are related to problems from operations research, especially vehicle routing (e.g., see [19]).

Searching and exploration have been extensively investigated in numerous settings. Using collections of mobile agents for the tree environment, the previous papers attempted to optimize the usage of various resources, e.g., the time for exploration (the maximal time used among all agents)

[6,16], memory used by agents [4], the number of agents [15], etc. Many papers assumed no knowledge of the tree, which leads to distribute algorithms optimizing the competitive ratio (e.g. [15,17]). In the centralized setting, the minimization of maximal time used by mobile agents is an NP-hard problem, even if the tree is a star and the collection contains only two robots (cf. [6,16]).

The task of broadcast is useful, e.g., when a designated leader needs to share its information with collaborating agents in order to perform together some future tasks. The broadcast problem for stationary processors has been extensively studied both in the case of the message passing model, (e.g. [7]), and for the wireless model, (see [10]).

The question of energy awareness has been investigated in different contexts. Paper [2] studied power management of (not necessarily mobile) devices. Several methods have been proposed to reduce energy consumption of computer systems including power-down strategies (see [2,5,18]) or speed scaling (cf. [20]). However, most research related to energy efficiency attempts to optimize the total power used in the entire system. When the power assignments concern the individual system components (as is the case of our model), the related optimization questions (e.g. see [8]) have a flavour of load balancing.

The communication problem by mobile agents has been studied in [1]. The agents of [1] perform efficient *convergecast* and *broadcast* in line networks. All agents of [1] have energy sources of the same size, allowing to travel the same distance. However, in the case of tree networks, the problems of convergecast and broadcast are proven to be strongly NP-complete in [1].

In the case, where agents may have different initial energy levels, [13] investigated a simpler communication problem of *data delivery*, when the information has to be transmitted between two given network nodes. This problem is proven to be NP-complete in [13] already for line networks. [11] showed that the situation is quite different when the agents are required to return to original locations. [11] gave a polynomial solution for data delivery in trees by returning agents. The problem of energy-efficient data delivery between given set of pairs of graph nodes was investigated in [12]. The data delivery and convergecast for trees with energy-exchanging agents were studied in [14], where linear-time solutions for both problems were proposed.

2 Agents starting from the source node

We start with an easier case when the root r is the same as the source node s, which contains the initial data packet. Observe that, even if we have unlimited number of agents in r, the problem is nontrivial. In the proposed solution, every agent i initially takes an exact amount of energy needed to traverse some subtree $T(i)$ and its traversal of $T(i)$ must be optimal. The union of all subtrees must sum up to the entire tree T and the choice of the subtrees must minimize the total energy needed to traverse them.

We consider separately cases of limited and unlimited number of agents. We will show that not all the agents are always activated, i.e. in some cases making walk too many agents would result in a suboptimal algorithm. We say that an agent is *activated* if it is used for walking (consumes a non-zero amount of energy), copies a data packet to its memory, when arriving to a node at which a copy of data packet is present, and subsequently disseminates it to all nodes visited afterwards.

Denote by T_v the subtree of T rooted at v.

Lemma 1. *Suppose that the source node s is the same as the root r. In every optimal broadcast algorithm, each activated agent should terminate its walk at a leaf of T.*

Proof. The proof goes by contradiction. Suppose that, in an optimal broadcast, some agent i terminates its walk in a non-leaf node v, traversing some edge (w, v) as the last edge of its route. Two cases are possible:

Case1: The traversal of the last edge (w, v) does not coincide with the first visit of node v by agent i. In this case we can remove the traversal of the last edge (w, v) from the route of agent i and the tree explored by agent i remains the same. However, such shortening of the route of agent i reduces its energy cost by $weight(w, v)$, which contradicts the optimality of the original traversal.

Case2: The traversal of the last edge (w, v) by agent i coincides with its first visit of node v. In such a case, agent i could not previously enter the subtree T_v (otherwise this would imply the second visit of v). Consequently, as T_v contains at least one leaf, unvisited by agent i, it must be visited by some other agent j. However, to reach any leaf of T_v from the starting position r, agent j must visit v on its route. As v does not need to be visited by two different agents, we can then again shorten the route of i by the last edge (w, v), reducing its cost. This contradicts optimality of its original route. □

The subset of leaves of T, at which the activated agents of an optimal algorithm terminate their paths, are called *critical leaves*. Each path from root r to a critical leaf is called a *critical path*. The union of all critical paths forms a tree, rooted at r, that we call the *frame* of the algorithm.

2.1 Scheduling agent movements when critical leaves are known

We start with the presentation of an algorithm which designs the movements of the agents once the set of critical leaves has been obtained. As agents possess the information about the packet from the start, it is sufficient to generate the trajectories of all robots, disregarding synchronization between actual movements of different agents.

Consider a subset $\mathcal{L}$ of critical leaves of tree T. Define $frame(\mathcal{L})$ as the union of all critical paths, i.e. the subtree of T induced by $\mathcal{L}$ and all its ancestors, see Figure 1. By $|frame(\mathcal{L})|$ we understand the sum of weights of all edges of $frame(\mathcal{L})$. Observe that the edges $T \setminus frame(\mathcal{L})$ form a set of subtrees rooted at the nodes of $frame(\mathcal{L})$. We call them *hanging subtrees* and we denote the set of hanging subtrees by $H(\mathcal{L})$.

Once we know the optimal set of critical leaves $\mathcal{L}$, then an optimal schedule is easy to construct. Below we give the algorithm ConstructSchedule, which constructs the optimal schedule for the given set of k agents. In fact we concentrate later only on computing the optimal $\mathcal{L}$ (needed in line 1 of the algorithm ConstructSchedule). Our main result is the computation of minimum cost in almost linear time, which also implies computing the optimal set $\mathcal{L}$.

```
Algorithm ConstructSchedule($k$);
1. Compute the set of critical leaves $\mathcal{L}$, such that $|\mathcal{L}| \leq k$, which maximizes $\Delta(\mathcal{L})$.
2. Assign to every critical leaf $l_i$ a different agent $i$ which will terminate its walk at $l_i$.
3. Assign arbitrarily each subtree $T' \in H(\mathcal{L})$ to a single critical leaf $L(T')$,
   such that $T'$ has the root on the critical path from $r$ to $L(T')$.
4. for each leaf $l_i \in \mathcal{L}$ do
     4.1. Agent $i$ follows the critical path from $r$ to the critical leaf $l_i$,
     4.2. On the way to its assigned critical leaf $l_i$ the agent $i$ makes a full DFS
          traversal of each hanging subtree $T' \in H(\mathcal{L})$ such that $L(T') = l_i$.
```

Observe that the total number of edge traversals, generated by the algorithm ConstructSchedule, can be quadratic.

Denote by $path(u, v)$ the set of nodes on the simple path between u and v (including u, v) and let $|path(u, w)|$ denote the distance (sum of edge weights) from node u to w in tree T. We denote also $depth(w) = |path(r, w)|$.

We define below a function $\Delta(\mathcal{L})$ which measures the efficiency of the broadcasting algorithm having $\mathcal{L}$ as its critical leaves.

$$\Delta(\mathcal{L}) \;=\; 2|frame(\mathcal{L})| - \sum_{w \in \mathcal{L}} depth(w) \tag{1}$$

The following lemma shows what is the value of $MinCost(T, k)$ - the energy cost of the schedule produced by the algorithm ConstructSchedule for k agents starting at the root of tree T. The energy depends on the choice of the set of critical leaves $\mathcal{L}$. The construction of the set $\mathcal{L}$ minimizing the energy cost will be discussed in the subsequent sections.

Lemma 2. *Assume k agents are placed initially in the source node $r = s$ of T. Then*

$$MinCost(T, k) \;=\; 2|E| - \Delta(\mathcal{L}),$$

where $\mathcal{L}$ is a subset of leaves maximizing $\Delta(\mathcal{L})$ over $|\mathcal{L}| \leq k$.

Proof. The algorithm has enough agents, so that to every critical leaf l_i corresponds a different agent i, which terminates its walk at l_i. The edges of all hanging subtrees $H(\mathcal{L})$, i.e. all edges of $T \setminus frame(\mathcal{L})$, are traversed twice in step 4.2 of the algorithm. Moreover each edge of a critical path is traversed in step 4.1 as many times as there are critical paths containing this edge. Consequently, the total cost of such traversal of T is twice the sum of lengths of edges belonging to $T \setminus frame(\mathcal{L})$, and the sum of the critical path lengths of the $frame(\mathcal{L})$. Hence the total cost equals

$$2|T| \setminus frame(\mathcal{L})| + \sum_{w \in \mathcal{L}} depth(w) \;=\; 2|E| \;-\; (2 \cdot |frame(\mathcal{L})| - \sum_{w \in \mathcal{L}} depth(w)) \;=\; 2|E| - \Delta(\mathcal{L}) \tag{2}$$

By Lemma 1, each optimal algorithm using at most k agents, corresponds to $frame(\mathcal{L})$ for some $\mathcal{L}$. Therefore, the cost represented in equation 2 is minimized for maximal $\Delta(\mathcal{L})$. □

Consequently, the broadcasting problem reduces in this case to the computation of $\mathcal{L}$ which maximizes $\Delta(\mathcal{L})$ with $|\mathcal{L}| \leq k$. The set $\mathcal{L}$ will be computed incrementally and in a greedy way. We conclude this section with some observations needed for the incremental construction of the optimal set of critical leaves.

Assume $\mathcal{L}$ is a set of leaves and consider a leaf $w \notin \mathcal{L}$. Denote by $LCA(w, \mathcal{L})$ the lowest common ancestor of w and some leaf from $\mathcal{L}$ (i.e. the lowest node belonging to $path(w, r)$ and $frame(\mathcal{L})$). Define $LCA(w, \emptyset) = r$. Let

$$\delta(w, \mathcal{L}) \;=\; |path(u, w)| - |path(r, u)|, \tag{3}$$

where $u = LCA(w, \mathcal{L})$. Equivalently we have

$$\delta(w, \mathcal{L}) \;=\; depth(w) - 2 \cdot depth(LCA(w, \mathcal{L})) \tag{4}$$

Observation 1 *For $\mathcal{L}_1, \mathcal{L}_2$ such that $\mathcal{L}_1 \subseteq \mathcal{L}_2$ and for any leaf w we have $\delta(w, \mathcal{L}_1) \geq \delta(w, \mathcal{L}_2)$.*

Indeed the statement of the Observation 1 follows from the fact that $\mathcal{L}_1 \subseteq \mathcal{L}_2$ implies $depth(LCA(w, \mathcal{L}_1)) \leq depth(LCA(w, \mathcal{L}_2))$.

Lemma 3. *For a given subset of leaves $\mathcal{L}$ and a leaf $\tilde{w} \notin \mathcal{L}$:*

$$\varDelta(\mathcal{L} \cup \{\tilde{w}\}) \;=\; \varDelta(\mathcal{L}) \,+\, \delta(\tilde{w}, \mathcal{L}). \tag{5}$$

Proof. If we add $\tilde{w}$ to $\mathcal{L}$, the new path between $LCA(\tilde{w}, \mathcal{L})$ and $\tilde{w}$ is added to $frame(\mathcal{L})$ (cf. Figure 1). Hence, according to formula 1 we have

$$\varDelta(\mathcal{L} \cup \{\tilde{w}\}) - \varDelta(\mathcal{L}) = 2|frame(\mathcal{L} \cup \{\tilde{w}\})| - 2|frame(\mathcal{L})| - \sum_{w \in (\mathcal{L} \cup \{\tilde{w}\})} depth(w) + \sum_{w \in \mathcal{L}} depth(w) =$$

$$= 2|path(LCA(\tilde{w}, \mathcal{L}), \tilde{w})| - depth(\tilde{w}) = depth(\tilde{w}) - 2 \cdot depth(LCA(\tilde{w}, \mathcal{L}) = \delta(\tilde{w}, \mathcal{L}) \qquad \square$$

2.2 A schematic algorithm computing the minimal cost

The formula (5) from Lemma 3 is used to design our algorithm Schematic-MinCost. The idea of the algorithm may be viewed as an incremental, greedy construction of the optimal set of critical leaves $\mathcal{L}$, by adding them, one by one. At each step we have a current version of the frame, which is augmented by a new subpath when a new leaf is added to $\mathcal{L}$. Consider $frame(\mathcal{L})$ obtained from the leaves $\mathcal{L}$ assigned to the first $i-1$ agents (that terminate their paths at $i-1$ leaves of $\mathcal{L}$). In the i-th iteration of the main loop we try to decide what is the best use of the next available agent. The i-th agent will terminate its traversal at some leaf w_i of T, not yet present in $frame(\mathcal{L})$. Therefore, $frame(\mathcal{L} \cup \{w_i\})$ will contain some new subpath, disjoint with $frame(\mathcal{L})$, starting at some vertex of $LCA(w_i, \mathcal{L})$ and ending at w_i. Observe, that the usage of agent i, permits the subpath from $LCA(w_i, \mathcal{L})$ to w_i to be traversed once (by a new agent) rather than twice (by some other agent which would need to perform a complete traversal of some subtree containing this path), which results in some energy gain. However such energy benefit is at the expense of bringing the agent from the root r to $(LCA(w_i, \mathcal{L})$. The main loop executions continue as long as such gain is possible (i.e. benefit minus expense is positive) and there are still available agents to be used. Such benefit is represented by the function $\delta(w_i, \mathcal{L})$ and our algorithm chooses the leaf offering the largest benefit. We prove later that this greedy approach results in construction of the best possible set of critical leaves.

```
Algorithm Schematic-MinCost($T, k$);
1.  $\mathcal{L} := \emptyset$;
2.  while $|\mathcal{L}| \leq k$ and $\exists(\, w \notin \mathcal{L})\, \delta(w, \mathcal{L}) > 0$ do
3.      choose a leaf $w \notin \mathcal{L}$ with maximum $\delta(w, \mathcal{L})$;
4.      $\mathcal{L} := \mathcal{L} \cup \{w\}$;
5.  return $|2E| - \varDelta(\mathcal{L})$;
```

Example 1. Figure 1 illustrates the execution of one step of the algorithm. The set $\mathcal{L}$ contains leaves w_1, w_2. The value of $\Delta(\mathcal{L}) = 36 - 19 = 17$, cf. formula 1. Among the remaining leaves, the maximal benefit is obtained by including w_4 in the set of critical leaves as $\delta(w_4, \mathcal{L}) = 3$. Then $\Delta(\{w_1, w_2, w_4\}) = 20$. As for the remaining leaves the values of δ are not positive, only three agents are activated (even if more are available) and, by Lemma 2, the cost of the optimal algorithm equals

$$2|E| - \Delta(\{w_1, w_2, w_4\}) = 56 - 20 = 36.$$

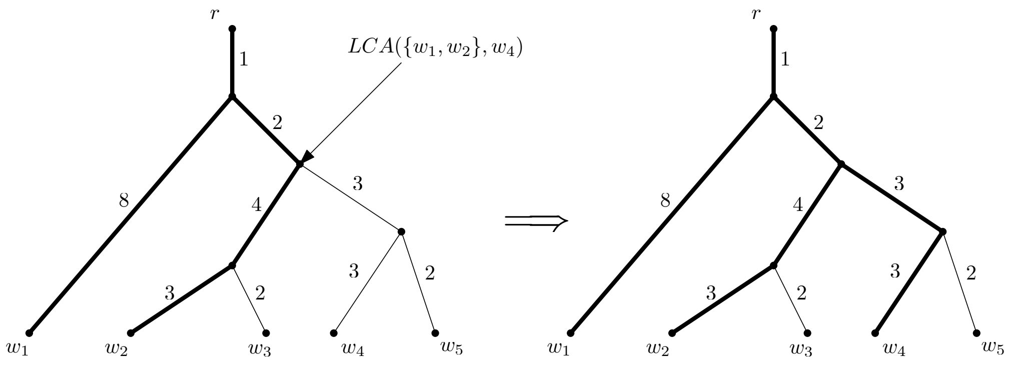

Fig. 1. The iteration which starts with a set of leaves $\mathcal{L} = \{w_1, w_2\}$, then w_4 is added to $\mathcal{L}$. Bold edges belong to the frames (subtrees $frame(\mathcal{L})$ of paths from the root to set $\mathcal{L}$ of critical leaves) before and after inclusion of w_4. We have $\Delta(\{w_1, w_2, w_4\}) = \Delta(\{w_1, w_2\}) + \delta(w_4, \{w_1, w_2\})$

Observe that algorithm Schematic-MinCost is in fact non-deterministic as it is possible that more than one leaf having the same value of δ may be chosen in line 3. Moreover, among optimal broadcasting algorithms, it is possible that the number of agents used may be different. This is possible if we activate an agent terminating at a leaf w for which $\delta(w, \mathcal{L}) = 0$.

Lemma 4. *Assume that in the algorithm Schematic-MinCost we insert the sequence $w_1, w_2, \dots w_m$ of leaves into $\mathcal{L}$ and there is a set $\mathcal{L}'$ maximizing $\Delta(\mathcal{L}')$ such that $\{w_1, w_2, \dots w_t\} \subseteq \mathcal{L}'$, $t < m$. Then there exists a set $\mathcal{L}''$, also maximizing $\Delta(\mathcal{L}'')$, which contains $\{w_1, w_2, \dots w_t, w_{t+1}\}$.*

Sketch of the proof. The idea of the proof is to show that the set $\mathcal{L}'$ must contain some leaf w^*, such that $w^* \neq w_i$, for $i = 1, 2, \dots, t+1$ and that exchanging w^* by w_{t+1} in the set $\mathcal{L}$" will not increase the cost of the corresponding broadcasting algorithm.

Lemma 5.
(a) *The set $\mathcal{L}$ computed by the algorithm Schematic-MinCost maximizes $\Delta(\mathcal{L})$.*
(b) *The value $2|E| - \Delta(\mathcal{L})$, output by the algorithm Schematic-MinCost, is the minimum amount of energy needed for broadcasting using k agents initially placed in the source $r = s$.*

Proof.
(a) Using inductively Lemma 4 we prove that the entire set $w_1, w_2, \dots w_m$ belongs to a set of critical leaves used by an optimal algorithm. By the exit condition of the while loop at line 2 of the algorithm Schematic-MinCost, there is no other leaf which may be added to such critical set of leaves improving the cost of the algorithm.

(b) This point follows directly from (a) and Lemma 2. □

Observe that every agent possesses the information about the packet at the very beginning of the algorithm. Then, as observed before, once the trajectories of each activated agent are determined, the timing of the travel of each agent is independent of the timing of the travel of any other agent. We conclude then by the following observation, which will be useful in the next section.

Observation 2 *Any energy-optimal schedule may be designed in such a way that the time intervals, during which agents perform their travel, are pairwise disjoint. In particular, we can choose any agent and make this agent complete its walk before any other agent starts walking.*

2.3 Efficient implementation of algorithm Schematic-MinCost

Efficiency of Schematic-MinCost depends on the cost of computing *on-line* the best $\delta(w, \mathcal{L})$. We replace it by introducing a more efficient function Gain(v) which does not depend on $\mathcal{L}$ and can be computed *off-line* in linear time. The algorithm Schematic-MinCost subsequently adds leaves to the set $\mathcal{L}$, each time choosing the leaf w offering the largest gain, i.e. the largest reduction $\delta(w, \mathcal{L})$ in the cost of the broadcasting schedule. The values of function δ for any leaf w, which does not yet belongs to $\mathcal{L}$, may change with subsequent modifications of $frame(\mathcal{L})$. In order to avoid recalculations of the function δ we propose the following solution.

Consider the moment when the leaf w is being added to the current set $\mathcal{L}$. Let v be a child of $LCA(w, \mathcal{L})$, which belongs to the path from $LCA(w, \mathcal{L})$ to w. Let $maxpath(v)$ be the longest path starting at v (and going away from the root). If there is more than one such path, we choose any one of them arbitrarily. We denote by leaf$(maxpath(v))$ the last node on such path. Observe that, at the moment when w is being added to $\mathcal{L}$, we have $|maxpath(v)| = |path(v, w)|$. For any node $v \neq r$ we define

$$\text{Gain}(v) \;=\; |maxpath(v)| + \text{weight}(\text{parent}(v), v) - |path(r, \text{parent}(v))| \qquad (6)$$

By convention, we also set Gain$(r) = |maxpath(r)|$.

Observation 3 *Assume $\mathcal{L}$ is a set of leaves. It follows from Equation 3, that*

$$\max\{\text{Gain}(v) : v \notin frame(\mathcal{L})\} = \max\{\delta(w, \mathcal{L}) : w \notin \mathcal{L}\}$$

Following the above Observation, in our algorithm we will be looking for nodes v, which are not in the current $frame(\mathcal{L})$.

```
Algorithm MinCostLimited($T, k$);
1.  $X := \{v \in V \,:\, \text{Gain}(v) > 0\}$;
2.  Sort $X$ with respect to Gain($v$) in non-increasing order;
3.  $\Delta := 0$ ; $\mathcal{L} := \emptyset$;
4.  while $X \neq \emptyset$ and $|\mathcal{L}| \leq k$ do
5.      choose $v \in X$ with maximum Gain($v$);
6.      $\Delta := \Delta + \text{Gain}(v)$;
7.      remove from $X$ all nodes belonging to $maxpath(v)$;
8.      $\mathcal{L} := \mathcal{L} \cup \text{leaf}(maxpath(v))$;
9.  return $2 \cdot |E| - \Delta$
/* $|\mathcal{L}|$ equals the number of activated agents */
```

Theorem 1. *The algorithm MinCostLimited(T, k) correctly computes in $O(n \log n)$ time the minimal amount of energy, which is needed to perform the data broadcast by k agents.*

Proof. We prove, by induction on the iteration of the while loop from line 4, that the node v chosen in line 5 does not belong to the current $frame(\mathcal{L})$. Indeed, in the first iteration of the while loop from line 4, $frame(\mathcal{L})$ is empty. In every other iteration, because of the leaf added to $\mathcal{L}$ in line 8, $frame(\mathcal{L})$ is augmented by the nodes of $maxpath(v)$, but all these nodes are then removed from set X in line 7. Therefore, in each execution of line 5 no node of X belongs to $frame(\mathcal{L})$.

Consequently, by Observation 3, every value of Gain chosen in line 5 of algorithm MinCostLimited is the same as the value of δ from the corresponding iteration of line 3 of algorithm Schematic-MinCost. Moreover, the same leaf is added to the set of critical set of leafs $\mathcal{L}$ in the corresponding iterations of both algorithms. The final critical set of leaves is then the same for both algorithms.

In the variable Δ is accumulated the sum of the values of function Gain for all nodes chosen in all iterations of the while loop. By Observation 3, after exiting the while loop, Δ equals the sum of values of function δ for all leafs from the final critical set $\mathcal{L}$. By Lemma 3, this sum equals $\Delta(\mathcal{L})$ and the final value of the computed cost equals $2|E| - \Delta(\mathcal{L})$. By Lemma 5 this proves the correctness of algorithm MinCostLimited.

We consider now the time efficiency of the algorithm. Observe first, that in the preprocessing, the values of Gain(v) can be computed in linear time. Recall that, by formula 6, we need to compute the values of $|maxpath(v)|$, weight$(\text{parent}(v), v)$ and $|path(r, \text{parent}(v))|$ Observe, that all these values may be computed using depth-first-search traversal (DFS) of T. Indeed weight$(\text{parent}(v), v)$ and $|path(r, \text{parent}(v))|$ may be obtained when DFS enters node v from its parent. On the other hand, $|maxpath(v)|$ is obtained when DFS visits v for the last time (arriving from its last child).

The amortized complexity of line 7 is also linear. Assume that X is implemented as a bidirectional list and each node v of the tree T contains a pointer to the element of X corresponding to Gain(v). Then the removal operation in line 7 takes constant time for each considered node v, hence the $O(n)$ time overall. As each other instruction inside the while loop takes constant time, the complexity of all lines of the algorithm, except line 2, is $O(n)$. The overall complexity is then dominated by the $O(n \log n)$ sorting in line 2. □

2.4 Unlimited number of agents in the source

For a set of nodes Y denote by children(Y) the set of all children of nodes in Y.

```
Algorithm MinCostUnlimited($T$);
/* The number of agents is unlimited */
1. $X := \{r\};\ \Delta := 0;\ \mathcal{L} := \emptyset$;
2. while $X \neq \emptyset$ do
3.     $v :=$ Extract any element of $X$;
4.     Add to $X$ each $x \notin maxpath(v)$ such that
           $parent(x) \in maxpath(v)$ and Gain$(x) > 0$;
5.     $\Delta := \Delta +$ Gain$(v)$;  $\mathcal{L} := \mathcal{L} \cup \{leaf(maxpath(v)\}$;
6.  / * $X = \{ v \in children(frame(\mathcal{L})) : v \notin frame(\mathcal{L}) \}$ */
7.  return $2 \cdot |E| - \Delta$
```

Theorem 2. *Assume that the number of agents initially placed at the source node is at least equal to the number of leaves of T. Then algorithm MinCostUnlimited(T) correctly computes in $O(n)$ time the minimal amount of energy needed for the broadcast in T.*

The idea of the proof is based on the fact that the set X is now restricted only to the children of the current frame and we choose each of them at some time. Since any two nodes, which are present in X at a same moment, never interfere (i.e. choosing one of them does never affect the Gain function of the other one), they may be treated in any order (as the number of available agents is sufficient for taking each of them at some time). This allows to avoid sorting and the time of the entire treatment is proportional to the number of edges of T.

3 All agents start from the same node r different from the source s

In this section we extend the consideration to the case when the initial position of the packet is not at the root of the tree. We show that this setting may be reduced to the case studied in the previous sections.

It would be helpful if we design the schedule, so that a robot moves along its trajectory independently from the timing of the motion of any other robot. By Observation 2, this was possible when the robots were initially placed at the source node s. However, in the current setting, at every time moment the robots executing an optimal schedule can be divided into two categories: the robots which already know the packet and the robots that do not. Clearly, the former category of robots are not restricted by their movement. On the other hand, the robots not knowing the packet might need to delay their movement as they may have to visit a node after the packet is deposited there.

The path between the root r and the source s we call the *backbone* of tree T and we denote it by B. We start with the following lemma.

Lemma 6. *There exists an optimal broadcasting algorithm in which the first activated agent starts moving towards the source node s, eventually returning to r, before any other agent is activated.*

Proof. The packet initially present at the source node s needs to be transported to all other nodes of the tree, including root r. Therefore, there must exist an agent which travels from r to s to pick up the packet. After that, a copy of the packet must be transported along the backbone B, starting at s and ending at r. During this travel of the packet along B, it may be transported by divers agents. However, when the packet is left by some agent i at a point p of B and picked later by some agent j, we can make agent i wait at point p until the arrival of agent j. At that moment, as agents are identical, we could exchange the roles of agents i and j and it is still agent i which continues to transport the packet. We conclude, by induction, that the packet is transported all the way by the same agent.

Observe as well, that the remaining agents that were exchanging roles with the agent i, in fact, do not need to start their travel before agent i reaches r. Indeed, they may wait at r until the packet is brought there by agent i and start their respective routes afterwards. □

We now construct the reduction from the setting where $r \neq s$ to the case $r = s$. For every instance I of the problem for $r \neq s$ we create an instance I' of the problem where $r = s$. We show that to solve I it is sufficient to solve I', where we use the results from the previous sections.

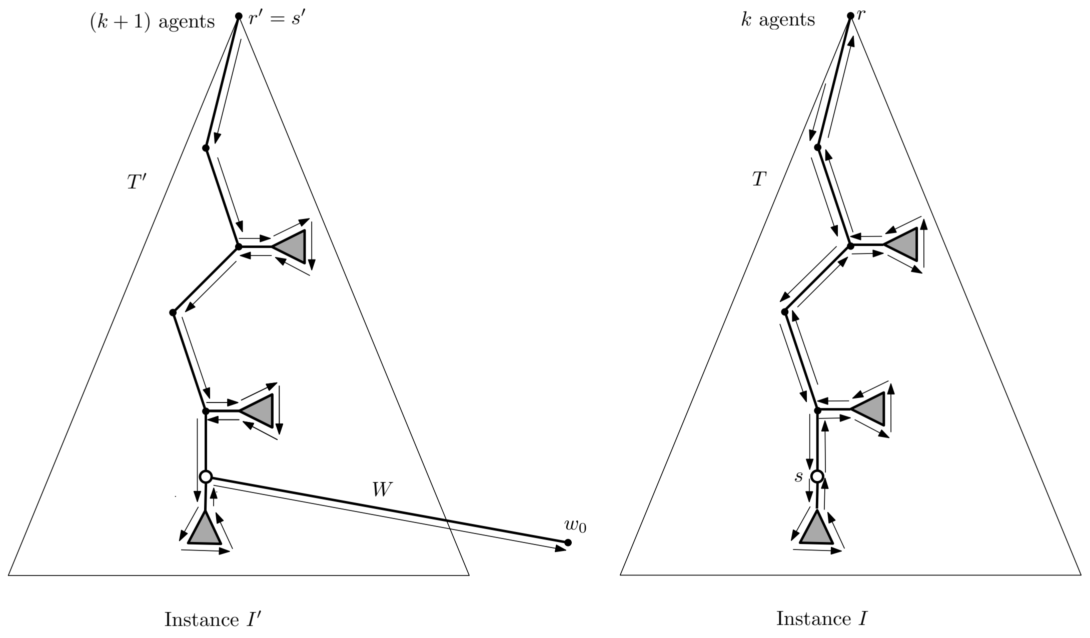

Fig. 2. On the left: the trajectory of the first activated agent, when the algorithm MinCostLimited is run for instance I'. The first agent terminates its walk in w_0. On the right: trajectory of the first activated agent for instance I. The first agent traverses the same nodes (except w_0) but returns to r to be reused later.

Let I be a given instance of the broadcast problem, in which we have tree T with k agents $1, 2, \ldots, k$ initially placed at root r and the source $s \neq r$.

We describe an instance I' of the broadcast problem with $s = r$ and $k + 1$ agents. We construct the tree T' by adding to T one extra leaf w_0 and an edge from node s to w_0 of weight W, where W equals the sum of weights of all edges of T. We define its root $r' = r$ in which we place $k' = k + 1$ mobile agents, represented by the integers $0, 1, \ldots, k$. We also set the source $s' = r'$ (see Fig. 2).

Lemma 7. [Reduction-Lemma] *If $s \neq r$ in T then*

$$MinCost(T, k) \;=\; MinCost(T',\, k+1) \,+\, |path(r, s)| \,-\, W,$$

where in T' the source node s' equals the initial location of $k + 1$ agents.

Proof. Consider first an optimal solution to the instance I' produced by algorithm ConstructSchedule. The weight of the edge incoming to node w_0 is so large that the leaf w_0 must belong to the set of critical leaves $\mathcal{L}$ and some agent 0 must terminate its walk in w_0. By Observation 2, we can suppose that agent 0 is the very first agent activated and the remaining agents didn't start before agent 0 reaches node w_0. Denote by T_H the set of all edges, traversed by agent 0, outside the simple path from r to w_0. By algorithm ConstructSchedule, T_H forms a subset of hanging subtrees.

Consider now an optimal solution to the instance I, which verifies Lemma 6. In this solution, the first activated agent 1, starting at r, travels along the backbone B to the source s (without any detour) and then continues its walk, eventually returning to r (bringing the packet), before any other agent starts moving. Obviously, on its way back along the backbone (i.e. from s to r) agent 1 may visit some nodes outside the backbone before returning to r.

Assume then, that agent 1, during its return from s to r along the backbone, traverses exactly the subtrees formed by the edges T_H (cf. Fig. 2). Consider the time moment in instance I' where agent 0 arrives at leaf w_0 and the time moment when in instance I agent 1 returns to root r. In both cases we have k agents and the packet available at the root r and the part of the tree that still needs to be explored equal to $T \setminus (T_H \cup B)$. Therefore, if we use the trajectories of the remaining k agents $1, 2, \ldots, k$ from the optimal solution of instance I' to complete the instance I the obtained solution of I is also optimal.

Observe that the assumption that agent 1 visited the subtrees formed by the edges of T_H may be dropped. Indeed, all subtrees of T_H are the hanging subtrees of the optimal solution and each of them is DFS traversed by some agent. Assigning any such subtree to agent 1 or any other agent visiting its root does not change the cost of the solution (recall line 3 of algorithm ConstructSchedule).

As from the moments when the situations in instances I' and I are identical all agents walk along the same trajectories in T and T', respectively, the cost of the solution of instance I differs from the solution of instance I' by the difference in the amounts of energy spent by the first agents of each instance, respectively. As this difference is $W - |path(r, s)|$, we have

$$MinCost(T, k) \;=\; MinCost(T', \, k+1) \,+\, |path(r, s)| \,-\, W$$

$\square$

Theorem 3. *Suppose that in the tree T the root r is different from the source s. We can solve the limited broadcast problem in $O(n \log n)$ time. If k is at least equal to the number of leaves in T we solve the broadcast problem in $O(n)$ time.*

Proof. Due to Lemma 7 limited broadcast reduces in linear time to the case when the source is the same as starting location of agents. In the unlimited case we can use Lemma 7 with $k = n$.

Hence the time complexity is asymptotically of the same order as that of the algorithm MinCostLimited(T, k), which is $O(n \log n)$. The case of unlimited broadcast can be done similarly in $O(n)$ time, by reduction to the algorithm MinCostUnlimited. $\square$

4 Final remarks

There are several open questions related to the communication problems and data delivery for mobile agents. One possible extension is to try to perform broadcast from a single source having mobile agents initially distributed in some nodes of the tree.

When the initial amounts of energy are a priori assigned to the agents, most communication problems for mobile agents are shown to be NP-complete (cf. [1, 13]). However, when the assignment of energy levels to the agents is left to the algorithm, minimization of total energy used for the communication problems remains open.

Another variation consists in broadcasting from a set of source nodes rather than a single one. If such set of sources involve all the tree nodes the problem becomes gossiping, in which the union of initial information of all nodes must reach every node of the network.

More general open question, which generalizes all communication protocols, concern the delivery of the union of information of a given set of nodes to another set of nodes. Finally, we can consider delivery from a set of specific sources to the respective specific target nodes. We believe that some variants of the question will lead to NP-hard problems for trees.

References

1. J. Anaya, J. Chalopin, J. Czyzowicz, A. Labourel, A. Pelc, Yann Vaxés: Collecting Information by Power-Aware Mobile Agents. *Proc. DISC* (2012), pp. 46-60.
2. S. Albers: Energy-efficient algorithms. *Comm. ACM 53(5)*, (2010), pp. 86-96.
3. S. Albers, M.R. Henzinger. Exploring unknown environments. *SIAM J. on Comput.*, (2000), 29 (4), pp.1164-1188.
4. C. Ambühl, L. Gasieniec, A. Pelc, T. Radzik and X. Zhang. Tree exploration with logarithmic memory, *ACM Trans. Algorithms*, (2011), 7 (4) pp. 17:1–17:21.
5. J. Augustine, S. Irani, C. Swamy. Optimal powerdown strategies. *SIAM J. Comput. 37* (2008), pp. 1499-1516.
6. I. Averbakh, O. Berman. A heuristic with worst-case analysis for minimax routing of two traveling salesmen on a tree. *Discr. Appl. Math.*, 68 (1996), pp. 17-32.
7. B. Awerbuch, O. Goldreich, D. Peleg and R. Vainish: A Trade-Off between Information and Communication in Broadcast Protocols. J. ACM 37(2): 238-256 (1990).
8. Y. Azar. On-line load balancing. In: *A. Fiat and G.Woeginger, Online Algorithms: The State of the Art, Springer* LNCS 1442, (1998), pp. 178-195.
9. R.A. Baeza-Yates, R. Schott: Parallel Searching in the Plane. *Comput. Geom. 5*, (1995), pp.143-154.
10. R. Bar-Yehuda, O. Goldreich, A. Itai: On the Time-Complexity of Broadcast in Multi-hop Radio Networks: An Exponential Gap Between Determinism and Randomization. J. Comput. Syst. Sci. 45(1): 104-126 (1992).
11. A. Bärtschi, J. Chalopin, S. Das, Y. Disser, B. Geissmann, D. Graf, A. Labourel and M. Mihalák. Collaborative Delivery with Energy-Constrained Mobile Robots. In *Proc. of SIROCCO* (2016), pp. 258-274.
12. A. Bärtschi, J. Chalopin, S. Das, Y. Disser, D. Graf, J. Hackfeld, P. Penna. Energy-Efficient Delivery by Heterogeneous Mobile Agents. In *Proc. of STACS* (2017), pp. 10:1-10:14.
13. J. Chalopin, R. Jacob, M. Mihalak, P. Widmayer: Data Delivery by Energy-Constrained Mobile Agents on a Line. *Proc. ICALP (2)*, (2014), pp. 423-434.
14. J. Czyzowicz, K. Diks, W. Rytter: Communication Problems for Mobile Agents Exchanging Energy *SIROCCO*, (2016), pp. 275-288.
15. S. Das, D. Dereniowski, C. Karousatou: Collaborative Exploration by Energy-Constrained Mobile Robots. In *Proc. of SIROCCO* (2015), pp. 357-369.
16. M. Dynia, M. Korzeniowski, C. Schindelhauer. Power-aware collective tree exploration. In *Proc. of ARCS* (2006), pp. 341-351.
17. P. Fraigniaud, L. Gasieniec, D. Kowalski, A. Pelc. Collective tree exploration. In *Proc. LATIN*, (2004), pp. 141-151.
18. S. Irani, S.K. Shukla, R. Gupta. Algorithms for power savings. *ACM Trans. on Algorithms*, Vol. 3, No. 4, Article 41, (2007).
19. P. Toth, D. Vigo. Vehicle routing: problems, methods, and applications. *SIAM*, (2014).
20. F.F. Yao, A.J. Demers, S. Shenker. A scheduling model for reduced CPU energy. In *Proc. of 36th FOCS* (1995), pp. 374-382.