Communication Problems for Mobile Agents Exchanging Energy

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

[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 mobile agents is deployed in the nodes of an edge-weighted network. Agents originally possess amounts of energy, possibly different for all agents. The agents travel in the network spending energy proportional to the distance traversed. Some nodes of the network may keep information which is acquired by the agents visiting them. The meeting agents may exchange currently possessed information, as well as any amount of energy. We consider communication problems when the information initially held by some network nodes have to be communicated to some other nodes and/or agents. The paper deals with two communication problems: data delivery and convergecast. These problems are posed for a centralized scheduler which has full knowledge of the instance. It is already known that, without energy exchange, both problems are NP-complete even if the network is a line. In this paper we show that, if the agents are allowed to exchange energy, both problems have linear-time solutions on trees. On the other hand for general undirected and directed graphs we show that these problems are NP-complete.

Keywords: mobile agents, data delivery, convergecast, energy exchange

Format: novel research contribution

1 Introduction

A set of n agents is placed at nodes of an edge-weighted graph G. An edge weight represents its length, i.e., the distance between its endpoints along the edge. Initially, each agent has an amount of energy (possibly distinct for different agents).

Agents walk in a continuous way along the network edges using amount of energy proportional to the distance travelled. An agent may stop at any point of a network edge (i.e. at any distance from the edge endpoints, up to the edge weight). Initially, at the nodes of the graph is stored the information (different for each node), which may be collected by agents visiting such nodes. Each agent has memory in which it can store all collected information.

When two agents meet, one of them can transfer a portion of currently possessed energy to another one. Moreover, two meeting agents exchange their currently possessed information, so that after the meeting both agents keep in their memories the union of the information previously held by each of them.

We consider two problems:

1. ***Data delivery* problem:** Given two nodes s, t of G, is it possible to transfer the initial packet of information placed at node s to node t?
2. ***Convergecast* problem:** Is it possible to transfer the initial information of all nodes to the same agent ?

We will look for schedules of agent movements which will not only result in completing the desired task, but also attempt to maximize the unused energy. We call such schedules *optimal*. We conservatively suppose that, whenever two agents meet, they automatically exchange the entire information they hold. This information exchange procedure is never explicitly mentioned in our algorithms, supposing, by default, that it always takes place when a meeting occurs.

1.1 Our results

We show that both communication have linear time algorithms on trees. On the other hand, for general undirected and directed graphs we show that these problems are NP-complete.

1.2 Related work

Recent development in the network industry triggered the research interest in mobile agents computing. Several applications involve physical mobile devices like robots, motor vehicles or various wireless gadgets. Mobile agents are sometimes interpreted as software agents, i.e., programs migrating from host to host in a network, performing some specific tasks. Examples of agents also include living beings: humans (e.g. soldiers or disaster relief personnel) or animals. Most studied problems for mobile agents involve some sort of environment search or exploration (cf. [3,8,10,11,12]). In the case of a team of collaborating mobile agents, the challenge is to balance the workload among the agents in order to minimize the exploration time. However this task is often hard (cf. [13]), even in the case of two agents in a tree, [6]. The tree exploration by energy-constrained mobile robots has been considered in [10].

The task of convergecast is important when agents possess partial information about the network (e.g. when individual agents hold measurements performed by sensors located at their positions) and the aggregate information is needed to make some global decision based on all measurements. The convergecast problem is often considered as a version of the data aggregation question (e.g.

[16,17]) and it has been investigated in the context of wireless and sensor networks, where the energy consumption is an important issue (cf. [4,15]).

The power awareness question has been studied in different contexts. Energy management of (not necessarily mobile) computational devices has been studied in [2]. To reduce energy consumption of computer systems the methods proposed include power-down strategies (see [2,5,14]) or speed scaling (cf. [18]). Most of research on energy efficiency considers optimization of overall power used. When the power assignments are made by the individual system components, similar to our setting, the optimization problem involved has a flavor of load balancing (cf. [7]).

The problem of communication by energy-constrained mobile agents has been investigated in [1]. The agents of [1] all have the same initial energy and they perform efficient convergecast and broadcast in line networks. However the same problem for tree networks is proven to be strongly NP-complete in [1].

The closely related problem of data delivery, when the information has to be transmitted between two given network nodes by a set of energy constrained agents has been studied in [9]. This problem is proven to be NP-complete in [9] already for line networks, if the initial energy values may be distinct for different agents. However, in the setting studied in [1,9], the agents do not exchange energy. In the present paper we show that the situation is quite different if the agents are allowed to transfer energy between one another.

2 The line environment

In this section we start with a line environnement and suppose that we are given a collection of agents $\{1, 2, \ldots, n\}$ on the line. Each agent i is initially placed at position a_i on the line and has initial energy e_i.

2.1 Data delivery on the line

We start with the delivery problem from point s to t. Assume that $a_i < a_j$ for $i < j$ and $s < t$. Indeed, in this case w.l.o.g. we may replace many agents starting at the same point may by a single agent holding the sum of their energy amounts.

The problem can be immediately reduced to the situation $s \leq a_1,\ a_n \leq t$. Otherwise, the agents on the left-hand side of s (starting from the leftmost one) walk left-to-right collecting energy of the encountered other agents. If some energy can be brought this way to s, we obtain an extra agent which will start at s. Symmetrically, the agents on the right-hand side of t act in order to possibly bring the maximal amount of energy to point t. It is easy to see that this is the best use of agents placed outside the interval $[s, t]$. Consequently, we may assume $s \leq a_1,\ a_n \leq t$.

Our first algorithm is only a decision version. Its main purpose is to show how certain useful table can be computed; all subsequent algorithms are based on computing similar type of tables.

Consider the partial delivery problem $\mathcal{D}_i$, in which agents larger than i are removed, together with their energy, and the goal is to deliver the packet from point a_1 to point a_i. We say that the problem $\mathcal{D}_i$ is *solvable* iff such a delivery is possible.

We define the following table $\overrightarrow{\Delta}$:

- If $\mathcal{D}_i$ is not solvable then $\overrightarrow{\Delta}_i = -\delta$, where δ is the *minimal* energy which needs to be added to e_i (to the energy of i-th agent) to make $\mathcal{D}_i$ solvable.

– If $\mathcal{D}_i$ is solvable then $\overrightarrow{\Delta}_i$ is the *maximal* unused energy which can remain in point a_i after delivering the packet from a_1 to a_i. Note that it is possible that $\overrightarrow{\Delta}_i > e_i$ since during delivery the unused energy of some other agents can be moved to point a_i.

Assume that points s and t are the starting points $s = a_0$ and $t = a_{n+1}$ of virtual dummy agents 0 and $n + 1$, respectively, each having zero energy. Therefore, we may assume that the original positions of the agents are $s = a_0 \leq a_1 < a_2 < a_3 < \ldots < a_n \leq t = a_{n+1}$.

We have the following decision algorithm.

ALGORITHM DELIVERY-TEST-ON-THE-LINE ;

```
1.  A := e_0 = 0; a_0 := s; a_{n+1} := t; e_{n+1} := 0;
2.  for i = 1 to n + 1 do
3.      d := a_i − a_{i−1};
4.      if A ≥ d then A := A − d
5.      else if A ≥ 0 then A := −2(d − A)
6.      else A := A − 2d;
7.      A := A + e_i; Δ→_i := A;
8.  return (A ≥ 0) ;
```

Example 1. Assume

$$[a_0, a_1, \ldots, a_4] \;=\; [0,\, 10,\, 20,\, 30,\, 40,\, 50],\; [e_0, e_1, \ldots e_4] \;=\; [0,\, 24,\, 10,\, 40,\, 0].$$

Then (assuming, by convention, $\overrightarrow{\Delta}_0 = 0$, see also Figure 1) we have

$$\overrightarrow{\Delta} \;=\; [0,\, 4,\, -2,\, 18,\, 8].$$

Remark. The values of $\overrightarrow{\Delta}_i$ are not needed to solve the decision-only version. However they will be useful in creating the delivery schedule and also in the *convergecast* problem.

Lemma 1. *The algorithm* DELIVERY-TEST-ON-THE-LINE *correctly computes the table* $\overrightarrow{\Delta}$ *(thus it solves the decision version of the delivery problem) in linear time.*

Proof. We prove by induction on i, that the value of $\overrightarrow{\Delta}_i$ is correctly computed in line 7 of the algorithm.

Suppose first the case $i = 1$. In the case $a_0 = a_1$, as $A = e_0 = 0$, in lines 4 and 7 we compute the value of $A = \overrightarrow{\Delta}_1 = e_1$, which is correct as agent 1 does not need to use any energy to pick up the packet at point a_0. Otherwise, if $a_0 < a_1$ we have $A = 0$ and $A < d$, so lines 5 and 7 are executed, in which case we have $A = \overrightarrow{\Delta}_1 = e_1 - 2d$. As agent 1 needs to cover distance d in both directions to bring the packet to point a_1 this is correct, independently whether the computed value negative or not.

Suppose now, by inductive hypothesis, that the algorithm computed correctly $A = \overrightarrow{\Delta}_{i-1}$ in the previous iteration. There are three cases:

Case 1 (line 4 of the algorithm). The instance $\mathcal{D}_{i-1}$ was solvable and after moving the packet from a_1 to a_{i-1} the maximal remaining energy was $\overrightarrow{\Delta}_{i-1}$. As in this case we have $\overrightarrow{\Delta}_{i-1} = A \geq d$, the energy $\overrightarrow{\Delta}_{i-1}$ is sufficient to move the packet from a_{i-1} to a_i. Consequently, we spent d energy to travers the distance d in one direction and we have $\overrightarrow{\Delta}_i = \overrightarrow{\Delta}_{i-1} - d + e_i$ as correctly computed in lines 4 and 7.

Case 2 (line 5). The instance $\mathcal{D}_{i-1}$ was still solvable but after moving the packet from a_1 to a_{i-1} the remaining energy $\overrightarrow{\Delta}_{i-1}$ is not sufficient to reach a_i without *help* from agents to the right of a_{i-1}. Then the $(i-1)$-st agent moves only one-way by distance $\overrightarrow{\Delta}_{i-1}$. The remaining distance $d - \overrightarrow{\Delta}_{i-1}$ to point a_i should be covered both-ways from a_i. Hence we need to use the amount of $2(d - \overrightarrow{\Delta}_{i-1})$ energy, which is expressed by statement 5. The value of $\overrightarrow{\Delta}_i$ is computed correctly independently whether the addition of e_i makes it positive or not.

Case 3 (line 6). In this case the instance $\mathcal{D}_{i-1}$ was not solvable, i.e. the agents $1, 2, \ldots, i-1$ could not deliver the packet to point a_{i-1}. Consequently, the interval $[a_{i-1}, a_i]$ has to be traversed entirely in both direction and we obtain $\overrightarrow{\Delta}_i = \overrightarrow{\Delta}_{i-1} - 2d + e_i$, which is correctly computed in lines 6 and 7.

The cases correspond to the statements in the algorithm, and show its correctness. This completes the proof.

Once the values of $\overrightarrow{\Delta}_i$ are computed, the schedule describing the behaviour of each agent is implicitly obvious, but we give it below for reference. Note that the action of each agent a_i is started once the process involving lower-numbered agents has been completed. We are not interested in this paper in finding the shortest time to complete the schedule (allowing agents to work in parallel).

ALGORITHM Delivery-Schedule-on-the-Line ;
{ Delivering packet from s to t }
$pos := s$;
for $i = 1$ to n **do**
  if $\overrightarrow{\Delta}_i \geq 0$ and $pos < a_i$ **then**
    1. The i-th agent walks left collecting energy of all encountered agents until arriving at the packet position. It picks up the packet.
    2. The i-th agent walks right collecting energy of all encountered agents until exhausting its energy or reaching t.
    3. The i-th agent leaves the packet at the actual position pos.

Delivery is successful iff $pos = t$;

Figure 1 illustrates the execution of the above algorithm for Example 1.

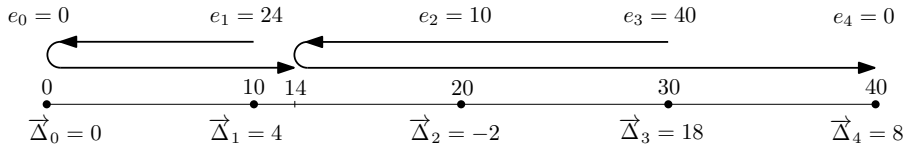

Fig. 1. Schedule of agent movements for a_i's and energies given in Example 1.

We conclude with the following theorem.

Theorem 1. *In linear time we can decide if the information of any agent can be delivered to any other agent and, if it is possible, we find the centralized scheduling algorithm which performs such a delivery.*

2.2 Convergecast on the line

The convergecast consists in communication in which the union of initial information of all nodes arrives to the same agent. The convergecast problem sometimes consists in verifying whether a given agent is a convergecast agent; other times, it has to be determined whether any such agent exists. For energy exchanging agents, if convergecast is possible then any agent may be its target, as agents may swap freely when meeting.

We present below the algorithm finding if convergecast is possible. We will use algorithm Delivery-Test-on-the-Line to compute the values of $\overrightarrow{\Delta}_i$ as defined before, assuming that $s = a_1$ and $t = a_n$. Similarly we denote by $\overleftarrow{\Delta}_i$ the values of the energy potential at point a_i that the symmetric algorithm would compute while transferring the packet initially situated at the point a_n towards the target position at a_i. Therefore, $\overleftarrow{\Delta}_i$ equals the deficit or the surplus of energy during the transfer of information initially held by agent n to agent i using agents $i, i+1, \cdots, n$.

ALGORITHM Convergecast-on-the-Line;

1. For all $i = 1, 2, \cdots, n$ compute the values of $\overrightarrow{\Delta}_i$ and $\overleftarrow{\Delta}_i$ representing the energy potentials at a_i, for deliveries from a_1 to a_i and a_n to a_i, respectively
2. **for** $i = 1$ to n **do**
3. **if** $\overrightarrow{\Delta}_i \geq 0 \wedge \overleftarrow{\Delta}_{i+1} \geq 0 \wedge \overrightarrow{\Delta}_i + \overleftarrow{\Delta}_{i+1} - (a_{i+1} - a_i) \geq 0$ **then**
4. **return** Convergecast possible;
5. **return** Convergecast not possible;

We have the following theorem.

Theorem 2. *Algorithm* Convergecast-on-the-Line *in $O(n)$ time solves the convergecast problem.*

Proof. The convergecast is possible if and only if the information of agent a_1 and the information of agent a_n may be transferred to the same point of the line. This is equivalent to the existence of a pair of agents i and $i+1$, such that transferring the information from point a_1 to a_i using agents $1, 2 \cdots, i$ results in a surplus of energy brought to point a_i, as well as that transferring the information from point a_n to a_{i+1} using agents $n, n-1 \cdots, i+1$ results in a surplus of energy brought to point a_{i+1}. Moreover, the sum of these two surpluses of energy must be sufficient to complete a walk along the entire segment $[a_i, a_{i+1}]$ permitting agents i and $i+1$ to meet. This is exactly what is verified at line 3 of algorithm Convergecast-on-the-Line.

An interested reader may observe, that the condition of the if clause from line 3 may be simplified to $\overrightarrow{\Delta}_i + \overleftarrow{\Delta}_{i+1} - (a_{i+1} - a_i) \geq 0$ as in such case the convergecast is also possible although the convergecast point may not be inside the interval $[a_i, a_{i+1}]$. However, the current condition at line 3 permits to identify all points of the environment to which the union of all node information may be transported. We call such points *convergecast points*. Indeed, if $\overrightarrow{\Delta}_i + \overleftarrow{\Delta}_{i+1} - (a_{i+1} - a_i) = 0$, then there exists a unique convergecast point inside the interval $[a_i, a_{i+1}]$. The surplus of energy permits to deliver the convergecast information to an interval of the line larger than a single point. We have the following Corollary.

Corollary 1. *If the condition in line 3 of algorithm* Convergecast-on-the-Line *is true, then the set of convergecast points of the line equals* $[a_{i+1} - \overleftarrow{\Delta}_{i+1}, a_i + \overrightarrow{\Delta}_i]$.

3 The tree environment

3.1 Data delivery in the tree

The technique developed for delivery in lines can be extended easily to delivery in undirected trees. In this case, the agents are placed at the nodes of the tree. Observe that from the original tree we can remove subtrees which do not contain s, t or any agents. Consequently, we obtain a connected tree whose every leaf either contains s or t or an initial position of some agent.

The delivery problem for a tree is easily reducible to the case of a line.

Theorem 3. *We can solve delivery problem and construct delivery-scenario on the tree in linear time.*

Proof. Consider the path π in the tree T connecting s with t. Suppose we remove from T all edges of path π. The tree splits into several subtrees *anchored* at nodes of π. For each such subtree we direct all edges towards the root, which is a node of π. The agents initially present at the leaves of such trees are walking up along the directed paths towards their roots accumulating energies at intermediate nodes. To avoid having two agents walking along the same edge it is sufficient to move agents present at leaves only and remove every such edge after the move is made. Agents having energy use it during their walk bringing the remainder to the intermediate nodes. Agents with zero energy are moved freely bringing no energy. The process terminates when the subtree is reduced to a single root belonging to path pi. This way we optimize the energy that can be brought to path pi. The problem of the delivery on the tree is now reduced to the delivery on the line π. Consequently, all steps of this construction may be computed in linear time. This completes the proof.

3.2 Convergecast on the tree

In this section we extend to the case of trees the basic ideas developed for the problem of convergecast for the line environment. The tables $\overleftarrow{\Delta}$ and $\overrightarrow{\Delta}$ for lines were computed locally, looking only at neighboring nodes. Simiarly, the values of the corresponding table $\overrightarrow{\Delta}$ for a node in a tree is computed looking at the neighbors of this node. However, as the flow of the information passing through node v can be made in d_v directions, where d_v is the degree of v, for each node v we will compute d_v different values of Δ . For this purpose, though the input tree is undirected, we will consider direction of edges. For each undirected edge (u, v) we consider two directed edges $u \to v$, $v \to u$. We define the subtree $T_{v\to u}$ as the connected component containing v and resulting by removing from T the edge (v, u), see Figure 2. Observe that at the moment of convergecast, there are two agents meeting at a point of some edge, that we call *convergecast point*, where these agents start possessing the initial information of all nodes.

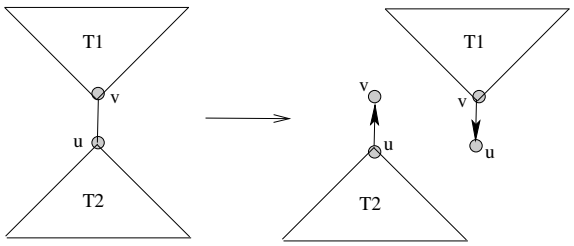

Fig. 2. Testing if there is a *convergecast point* on the undirected edge (u, v) is reduced to computation of the costs $\Delta_{u\to v}$ and $\Delta_{v\to u}$ of moving all packets in the trees $T2 = T_{u\to v}$ and $T1 = T_{v\to u}$.

In order to compute all needed values of Δ, for each directed edge $u \to v$ of the tree we define $\Delta_{u\to v}$ as the energy potential of moving all packets from the subtree $T_{u\to v}$ to its root u without interacting with any node outside $T_{u\to v}$. More exactly, if $\Delta_{u\to v} \geq 0$, then it represents the maximal amount of energy that can be delivered to u, together with all data packets originated at the nodes of $T_{u\to v}$. Observe that, if $T_{u\to v}$ initially does not contain any agents, then $\Delta_{u\to v}$ equals twice the sum of weights of all edges of $T_{u\to v}$. Indeed, in such case, the delivery must be performed by an agent starting at u and performing the DFS traversal of $T_{u\to v}$. If $T_{u\to v}$ initially contains some agents, the value of $\Delta_{u\to v}$ is smaller, but always equal at least the sum of weights of its edges. If $\Delta_{u\to v} < 0$ then $-\Delta_{u\to v}$ is the minimal amount of energy that we need to deliver to u by some agent, initially outside $T_{u\to v}$, so that this agent can bring to node u all data packets from the nodes of $T_{u\to v}$. In both cases, will be used all agents initially present inside $T_{u\to v}$ as well as their entire energy. We propose the following algorithm.

ALGORITHM Convergecast-on-the-Tree(T);

1. $X :=$ the set of directed edges of T;
2. **while** $X \neq \emptyset$ **do**
3. Choose $u \to v \in X$, such that for each $x \to u$, $x \neq v \implies x \to u \notin X$
4. Compute $\Delta_{u\to v}$; Remove $u \to v$ from X;
5. **for** each directed edge $(u, v) \in V$ **do**
6. **if** $(\Delta_{u\to v} \geq 0) \wedge (\Delta_{v\to u} \geq 0) \wedge (\Delta_{u\to v} + \Delta_{v\to u} \geq weight(u, v))$
7. **then return** Convergecast is possible
8. **return** Convergecast is not possible

The values of $\Delta_{u\to v}$ are computed by the following procedure.

PROCEDURE Compute $\Delta_{u\to v}$;

1. $\Delta_{u\to v} := e_u$; {initial energy of node u}
2. **for** each undirected edge $x \to u$, such that $x \neq v$ **do**
3. **if** $\Delta_{x\to u} \geq weight(x, u)$
4. **then** $\Delta_{u\to v} := \Delta_{u\to v} + \Delta_{x\to u} - weight(x, u)$
5. **else if** $\Delta_{x\to u} > 0$
6. **then** $\Delta_{u\to v} := \Delta_{u\to v} + 2 * (\Delta_{x\to u} - weight(x, u))$
7. **else** $\Delta_{u\to v} := \Delta_{u\to v} + \Delta_{x\to u} - 2 * weight(x, u)$

We have the following theorem:

Theorem 4. *Algorithm* Convergecast-on-the-Tree *in linear time solves the convergecast problem for trees.*

Proof. We show first the following claim:
Claim:, The **while** loop from line 2 of the algorithm calls the function Compute $\Delta_{u\to v}$ in line 4 for all undirect edges of T and the value of $\Delta_{u\to v}$ is correctly computed for every directed edge $u \to v$.

The proof of the claim goes by induction on the consecutive iterations of the **for** loop from line 2. Consider first node u which is a leaf in T. The terminal edge $u \to v$ of any such node may be taken in the first iteration of the **for** loop from line 2. As no iteration of the **for** loop from line 4

is ever executed for $u \to v$ and the tree $T_{u\to v}$ is composed of a single node, we obtain correctly $\Delta_{u\to v} = e_u$, i.e. the initial energy of node u.

Consider now a non-terminal node u. Let $v_1, v_2, \cdots, v_p$ be all nodes adjacent to u, such that $v_i \neq v$, for $i = 1, \cdots, p$. Note that at some point we have $(v_i, u) \notin X$, for all $i = 1, \cdots, p$ and the values of $\Delta_{v_i\to u}$ for all $i = 1, \cdots, p$ have been computed. Observe that, bringing packets from all $v_i \neq v$, $i = 1, \cdots, p$ to u needs to be done across the respective edges $v_i \to u$, sometimes bringing the unused energy to u and other times using some energy from $\Delta_{u\to v}$ to traverse twice edge (v_i, u), or its portion, by an agent coming from u.

Take any v_i and suppose first that $\Delta_{v_i\to u} \geq weight(v_i, u)$. Then by inductive assumption, the agents present at $T_{v_i\to u}$ can perform the convergecast to v_i bringing there the amount of $\Delta_{v_i\to u}$ extra energy. This energy is sufficient to transfer all packets of $T_{v_i\to u}$ through edge (v_i, u) and the remaining amount of $\Delta_{v_i\to u} - weight(v_i, u)$ energy is accumulated at $\Delta_{u\to v}$, which is correctly computed at line 4 of procedure Compute $\Delta_{u\to v}$.

Suppose now, that $\Delta_{v_i\to u} \leq 0$. In order to bring all packets of $T_{v_i\to u}$ to u, an agent present at u must traverse the edge $u \to v_i$, bring the packets to node using $-\Delta_{v_i\to u}$ extra energy and then traverse the edge (u, v_i) in the opposite direction $v_i \to u$. For this purpose is needed the extra energy of $-\Delta_{x\to u} + 2 * weight(x, u)$, which is correctly suppressed from $\Delta_{u\to v}$ at line 7 of the procedure.

Consider now the remaining case when $0 < \Delta_{v_i\to u} < weight(v_i, u)$. In this case, all packets of $T_{v_i\to u}$ are brought to node v_i by some agent initially present within $T_{v_i\to u}$, but this agent does not have enough energy to traverse edge $v_i \to u$ by itself. Such agent will use its entire energy of $\Delta_{v_i\to u}$ to traverse a portion of edge $v_i \to u$ and some other agent need to come from u and to traverse the other portion in both directions in order to transfer the packets to u. The energy needed by the second agent equals $2 * (weight(v_i, u) - \Delta_{v_i\to u})$, which is correctly suppressed from $\Delta_{u\to v}$ at line 6 of the procedure. This completes the proof of the claim.

To complete the proof, consider the moment when in an optimal convergecast algorithm one agent obtains the union of the initial information of all nodes of the network. This happens while two agents meet on some edge (u, v), one of them carrying the union of information from the subtree $T_{u\to v}$, and the other one - from the subtree $T_{v\to u}$. These agents need to have enough positive energy to meet within the edge (u, v). This is equivalent to the condition tested in line 6 of the algorithm.

The condition from line 6 of algorithm Convergecast-on-the-Tree permits to decide only if the convergecast is possible. However, similarly to the line case, an interested reader may observe that one can easily identify the set of all convergecast points. For this purpose we define the set of $D_{u,v}(d)$ containing a subset of points from the edges of T. Consider a point p and the simple path $\Pi(u, p)$ of T joining p with u. We define $p \notin D_{u,v}(d)$ if the path $\Pi(u, p)$ goes in the direction of edge (u, v) and its length exceeds d, i.e. $|\Pi(u, p)| > d$. All other points of T belong to $D_{u,v}(d)$. We have the following corollary.

Corollary 2. *If the condition in line 6 of algorithm* Convergecast-on-the-Tree *is true, then the set of all convergecast points of the tree equals* $D_{u,v}(\Delta_{u\to v}) \cap D_{v,u}(\Delta_{v\to u})$.

4 NP-Completeness for digraphs and graphs

We use the following NP-complete problem:

Integer Set Partition (ISP): Given set $X = \{x_1, x_2, \ldots, x_n\}$ of positive integer values verify whether X can be partitioned into two subsets with equal total sums of values.

We have the following theorem.

Theorem 5. *The delivery and convergecast problems are NP-complete for general directed graphs.*

Proof. Denote $E = \sum_{i=1}^{n} x_i$. Given an instance of the ISP problem, we construct the following graph G_X (see Figure 3). The set of $n+3$ nodes of G_X consists of three nodes s, t, a and the set of nodes $V = \{v_1, v_2, \ldots, v_n\}$. Each node v_i contains a single agent i having an initial energy $e = x_i$, for $i = 1, 2, \ldots, n$. The weights of edges outgoing from nodes of V are $w(v_i \to s) = x_i/3$ and $w(v_i \to a) = 0$ for $i = 1, 2, \ldots, n$. Moreover we have $w(s \to a) = E/3$ and $w(a \to t) = E/2$. Consider a delivery from s to t. W.lo.g. we can suppose that this is done by some agent i, which

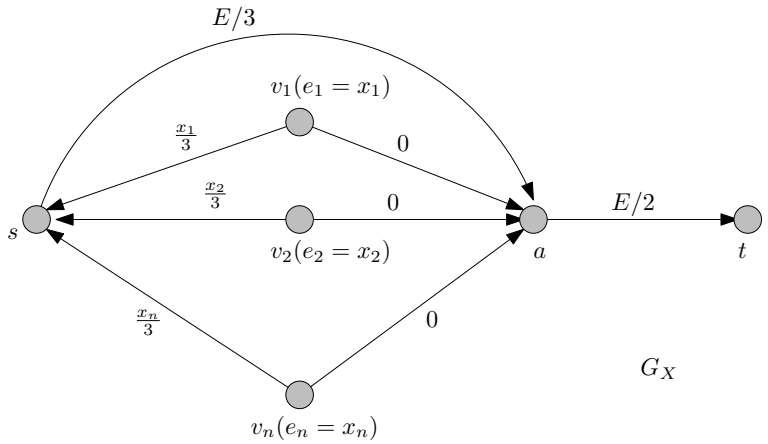

Fig. 3. Delivery from s to t is possible iff the set of weights x_i can be partitioned into two sub-sets of the same sum.

must traverse the path $v_i \to s \to a \to t$. As no agent can do it using only its own energy (otherwise $x_i \geq 5E/6$ and the ISP trivially has no solution), some other agents of the collection must walk to s and some other ones must go directly to a, in order to deliver to agent i additional energy needed to complete its path $v_i \to s \to a \to t$.

Assume that X_1, X_2 are the sets of agents which directly move to s and a, respectively. Let

$$\alpha \;=\; \sum_{i \in X_1} x_i, \;\; \beta \;=\; \sum_{i \in X_2} x_i$$

Hence the energy delivered to s, unused by the agents X_1 incoming to s, is $\frac{2}{3}\alpha$. As this energy must be sufficient to traverse at least edge $s \to a$, we have

$$\frac{2}{3}\alpha \;\geq\; E/3 \tag{1}$$

Consider now the maximal energy, which may be available to agent i at point a. It is equal to the sum of energy β, which is brought to point a by agents X_2, and the energy unused by agent i, ending its traversal of edge $s \to a$, which equals $2\alpha/3 - E/3$. As the sum of these energies must suffice to traverse edge $a \to t$ of weight $E/2$ and $\alpha + \beta = E$ we have

$$\frac{E}{2} \leq \beta + \frac{2}{3}\alpha - E/3 = \frac{1}{3}\alpha + \frac{2}{3}\beta = \frac{E}{3} + \frac{1}{3}\beta \tag{2}$$

From (1) we have $\alpha \geq E/2$ and (2) leads to $\beta \geq E/2$, which implies $\alpha = \beta = E/2$.

Consequently, the delivery from s to t in graph G_X is possible if and only if the given instance of the integer partition problem is solvable. This implies NP-completeness of the delivery problem.

As t is the only node having paths incoming from all other nodes, the convergecast for G_X implies the delivery from s to t, hence the convergecast problem is also NP-complete.

Theorem 6. *The delivery and convergecast problems are NP-complete for general undirected graphs.*

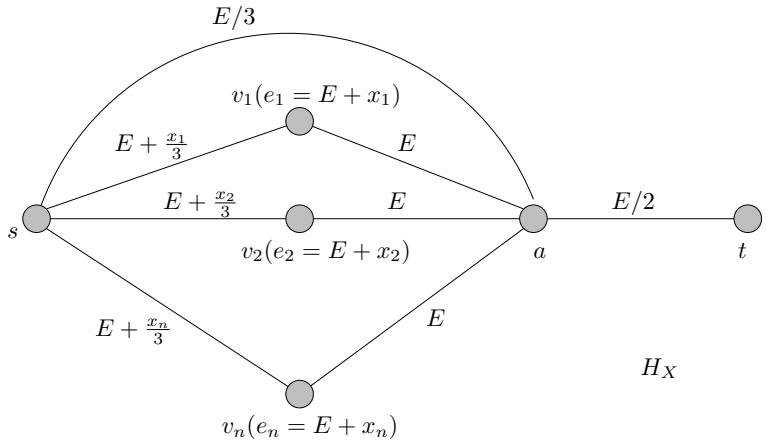

Fig. 4. The undirected version of the graph from Figure 3. The weights of nodes v_i and lengths of edges incident to these nodes are increased by E.

Proof. Consider graph H_X - an undirected version of the graph from the previous proof (see Figure 4). Increase the energy of every agent by E, i.e. agent i, initially placed at node v_i, now has energy $E + x_i$, for $i = 1, 2, \ldots, n$. Moreover increase by E the weight of each edge, which is incident to node v_i, i.e. $w(s, v_i) = E + x_i/3$ and $w(v_i, a) = E$, for $i = 1, 2, \ldots, n$.
Delivery. Consider delivery from s to t. Observe that no edge incident to v_i, for $i = 1, 2, \ldots, n$, can be used twice. Indeed, in order to transfer energies between agents they have to meet moving from their initial positions. However, at the moment of such meeting the sum of the remaining energies is smaller than E, which does not permit to traverse any edge incident to x_i for the second time. Clearly traversing directed edges $a \to s$ and $t \to a$ is also useless, hence the delivery from s to t in graph H_X is equivalent to the respective delivery in G_X.
Convergecast. If we consider t as the convergast node, the conergecast problem is equivalent to the delivery from s to t, which implies its NP-completeness.

5 Final Remarks

It is somewhat remarkable that, without energy exchange, even the simplest problem of data delivery is NP-complete in the simplest environment of the line, while, as we have shown in this paper, considered communication problems with energy exchange are solvable in linear time even for tree networks. On the other hand, it is not surprising that energy exchange in general graphs does not help and the problems are NP-complete.

There remain interesting open problems using energy-exchanging mobile agents for other communication protocols, like broadcast or gossiping. Observe that, in the case of data delivery and convergecast, each point of the tree is traversed at least once and at most twice (once in each direction) in the optimal solutions. However, in the case of broadcast, i.e. when the information of one node must be delivered to all other nodes of the tree, some tree edges need to be traversed by several agents. E.g., this is the case of weighted star with many agents starting at the same leaf. The problem of gossiping is even more involved.

An interested reader may try to extend the proposed solutions to the case when the data delivery needs to be performed from a given subset of nodes into another subset. Further possible extension is to realize one-to-one delivery using a configuration of energy-exchanging agents, i.e. when each of n source nodes must deliver to a specific target.

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.*, 29 (4), pp.1164-1188.
4. V. Annamalai, S. K. S. Gupta, and L. Schwiebert, On Tree-Based Convergecasting in Wireless Sensor Networks, *Wireless Communications and Networking, IEEE*, vol. 3 (2003), pp. 1942 - 1947.
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. 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.
8. R.A. Baeza-Yates, R. Schott: Parallel Searching in the Plane. *Comput. Geom. 5*, (1995), pp.143-154.
9. 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.
10. S. Das, D. Dereniowski, C. Karousatou. Collaborative Exploration by Energy-Constrained Mobile Robots. In *Proc. of SIROCCO* (2015), pp. 357-369.
11. M. Dynia, M. Korzeniowski, C. Schindelhauer. Power-aware collective tree exploration. In *Proc. of ARCS* (2006), pp. 341-351.
12. P. Fraigniaud, L. Gasieniec, D. Kowalski, A. Pelc. Collective tree exploration. In *Proc. LATIN*, (2004), pp. 141-151.
13. G. Frederickson, M. Hecht, C. Kim. Approximation algorithms for some routing problems. *SIAM J. on Comput.*, 7 (1978), pp. 178-193.
14. S. Irani, S.K. Shukla, R. Gupta. Algorithms for power savings. *ACM Trans. on Algorithms*, Vol. 3, No. 4, Article 41, (2007).
15. A. Kesselman and D. R. Kowalski, Fast distributed algorithm for convergecast in ad hoc geometric radio networks, *Journal of Parallel and Distributed Computing*, Vol. 66, No. 4 (2006), pp. 578-585.
16. L. Krishnamachari, D. Estrin, S. Wicker. The impact of data aggregation in wireless sensor networks. *ICDCS Workshops*, (2002), pp. 575-578.
17. R. Rajagopalan, P. K. Varshney, Data-aggregation techniques in sensor networks: a survey *Comm. Surv. and Tutorials*, Vol. 8, No. 4, (2006), pp. 48 - 63.
18. F.F. Yao, A.J. Demers, S. Shenker, A scheduling model for reduced CPU energy. In *Proc. of 36th FOCS* (1995), pp. 374-382.