Truthful Mechanisms for Delivery with Mobile Agents

Andreas Bärtschi, Daniel Graf, and Paolo Penna

ETH Zürich, Department of Computer Science, Switzerland
{andreas.baertschi, daniel.graf, paolo.penna}@inf.ethz.ch

February 27, 2017

Abstract

We study the game-theoretic task of selecting mobile agents to deliver multiple items on a network. An instance is given by m messages (physical objects) which have to be transported between specified source–target pairs in a weighted undirected graph, and k mobile heterogeneous agents, each being able to transport one message at a time. Following a recent model [Bärtschi et al. 2016], each agent i consumes energy proportional to the distance it travels in the graph, where the different rates of energy consumption are given by weight factors w_i. We are interested in optimizing or approximating the total energy consumption over all selected agents.

Unlike previous research, we assume the weights to be private values known only to the respective agents. We present three different mechanisms which select, route and pay the agents in a truthful way that guarantees voluntary participation of the agents, while approximating the optimum energy consumption by a constant factor. To this end we analyze a previous structural result and an approximation algorithm given in [5, 4]. Finally, we show that for some instances in the case of a single message ($m = 1$), the sum of the payments can be bounded in terms of the optimum as well.

1 Introduction

We study the *delivery* of physical objects (henceforth called *messages*) between different locations by autonomous agents. Recent technological progress allows the mass production of battery-powered robots such as drones which can be deployed outdoors to deliver goods without the interaction with humans. The main concern for algorithm-design thus focuses on an energy-efficient operation of these *agents*. The primary energy expense is often defined by the movements of the agents; in this paper we consider the energy consumption to be proportional to the distance traveled by an agent. We assume the agents to be heterogeneous in the sense of the agents having different rates of energy consumption.

Recent progress in minimizing the *total energy consumption* in *delivery problems* has been made assuming that the agents belong to the entity which wants to deliver the messages [5, 4]. For this scenario it has been shown that (i) there is a polynomial-time constant-factor approximation algorithm (with the approximation guarantee depending on the energy consumption rates) and that (ii) one can restrict him/herself to only consider scenarios where each message is delivered by a single agent, without loosing more than a factor of 2 compared to an optimum overall delivery (in which agents can handover a message to another agent). The latter result has been named the *Benefit of Collaboration*. In general, the problem is NP-hard to approximate even for a single agent.

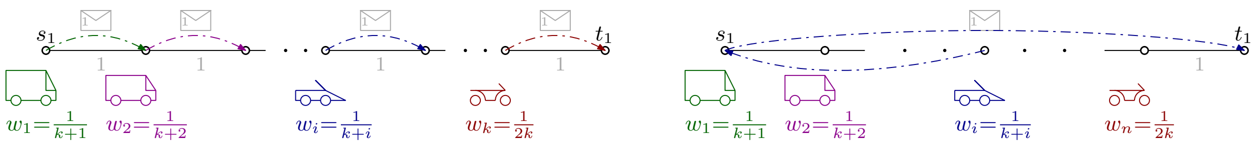


Figure 1: Delivery of a single message on a path of length k; k available agents of weights $w_i = \frac{1}{k+i}$. (left) Optimal solution using all agents; cost $= \sum_{i=1}^{k} w_i \cdot 1 = \sum_{i=1}^{k} \frac{1}{k+i} = \mathcal{H}_{2k} - \mathcal{H}_k \approx \ln 2$ $(k \to \infty)$. (right) Any solution using a single agent i has cost $w_i \cdot ((i-1)+k) = (k+i-1)/(k+i) \approx 1$ $(k \to \infty)$.

In the following, however, we assume the agents to be *independent selfish agents* which make a bid to transport messages (think e.g. about taxi drivers which make a bid to transport people via a carsharing-platform). In such a scenario, the rate of energy consumption is a private value known only to the respective agent. Our goal is hence to design a mechanism to select agents, plan their routes and reimburse them such that

(i) the overall spent energy is as close to the optimum as possible,
(ii) each agent announces its true rate of energy consumption in its bid, and
(iii) each agent is reimbursed at least the cost of energy it needs to deliver all of its assigned messages.

Our Model We are given a connected, undirected graph $G = (V, E)$ on $n := |V|$ nodes with the length of each edge together with m messages, each of which is given by a specified source-target node pair (s_j, t_j). Furthermore there are $k > 1$ mobile agents initially located at distinct nodes of the graph. Each agent i starts at position p_i and has a capacity of 1, meaning it can carry at most one message at a time. An agent can pick up a message at some node u and drop the message again at some other node v. In that sense it is also possible to hand over a message j between agents if agent a drops it at a node where another agent b later picks up the same message, in which case we say that the agents *collaborate* on message j. A scenario in which each message j is delivered from its source s_j to its target t_j is called a *feasible solution* of the delivery problem. Such a feasible solution x specifies the *travel distance* $d_i(x)$ that each agent i has to travel in this solution.

Each agent consumes energy proportional to the distance it travels in the graph, we are interested in optimizing the total energy consumption for the team of agents. Specifically, we consider heterogeneous agents with different rates of energy consumption (weights w_i), and therefore the energy spent by agent i in solution x is $w_i \cdot d_i(x)$, while the total energy of a solution is (see e.g. Figures 1,2):

$$\textsc{cost}(x, w) := \sum_{i=1}^{k} w_i \cdot d_i(x) \quad . \tag{1}$$

We denote by $\textsc{Opt}$ a solution with minimum cost among all solutions, $\textsc{Opt} \in \arg\min_x \textsc{cost}(x, w)$. It is natural to consider the scenario in which every agent is *selfish* and therefore cares only about its own cost (energy) and not about the optimum social cost (total energy).

Mechanism Design We consider the scenario in which agents can cheat or speculate about their costs: each w_i is a *private* information known to agent i who can report a possibly different cost. For instance, an agent may find it convenient to report a very high cost, in order to induce the underlying algorithm to assign a shorter travel distance to it (which may not be globally optimal). To deal with such situations, we introduce suitable compensations for the agents to incentivize them to truthfully report their costs. This combination of an algorithm and a payment rule is called a *mechanism*, and the following properties are desirable:

- *Optimality.* The mechanism runs some (nearly) optimal algorithm such that, if agents do not misreport, then the computed solution has an optimal (or nearly optimal) social cost $\textsc{cost}(x, w)$ with respect to the true weights w.

- *Truthfulness.* For every agent, truth-telling is a *dominant strategy*, that is, in no circumstance it is beneficial for an agent to misreport its cost. Meaning, no matter what all the other agents report, its utility (payment received minus its incurred cost for the computed solution) is maximized when reporting the true cost.

- *Frugality.* The total payment to the agents should be nearly optimal, that is, it should be comparable to the agents' costs. Since agents should be payed at least their cost when truthfully reporting (*voluntary participation*), the mechanism *must* pay a total amount of at least the optimum.

Intuitively speaking, we are paying the agents to make sure that they reveal their true costs, and, in this way, we can find a (nearly) optimal solution (wrt. sum of weighted travel distances). At the same time, we want to minimize our total payments, i.e., we would like not to spend much more than the actual cost (weighted travel distance) of the solution.

Our results After some preliminaries on mechanism design and on energy-efficient delivery in Section 2, we first investigate in Section 3.1 the constant-factor $(4 \cdot \max \frac{w_i}{w_j})$-approximation algorithm presented in [4]. We reason why this algorithm cannot be turned into a mechanism that is both truthful *and* guarantees voluntary participation. However, using the algorithm as a black box for a new algorithm A^* and applying Clarke's pivot rule for the payments to the agents, we give a truthful mechanism based on A^* which guarantees voluntary participation and incurs a total energy of at most $(4 \cdot \max \frac{w_i}{w_j})$ times the energy cost of an optimal delivery.

In Section 3.2 we consider instances where either the number of messages m is constant or the number of agents k is constant. For both cases we provide polynomial-time constant-factor approximation algorithms with approximation factors that are independent of the agents' weights. Both of these approximation algorithms rely on the Benefit of Collaboration [5]. Furthermore, they satisfy sufficient conditions to be turned into truthful mechanisms with voluntary participation. We show that the running time of the former algorithm can be improved to a yield a FPT-approximation mechanism, parametrized by the number of messages.

Finally, in Section 4, we discuss the frugality of mechanisms for the case of a single message ($m = 1$). In particular, we consider two truthful mechanisms, namely, the optimal one (which possibly uses several agents), and the one which always uses a single agent. Intuitively, though the latter mechanism has a higher cost, the total payment can be lower since we are paying one single agent (instead of several ones). We show that neither mechanism is, in general, better than the other in terms of total payments. However, under some assumptions on the input, the payments of both mechanisms are only a small multiplicative factor larger than the minimum necessary (the cost of the optimum). The bound for the mechanism with a single message relies on the Benefit of Collaboration [5].

Related work *Energy-efficient* delivery has not been studied until recently. To the best of our knowledge, most previous results are based on a model where the agents have uniform rates of energy-consumption but limited battery [7]. This restricts the possible movements of the agents –

one gets the decision problem of whether the given messages can be delivered without exceeding the available battery levels. This problem turns out to be NP-hard even for a single message [8, 3] and even if energy can be exchanged between the agents [10]. The model of unbounded battery but heterogeneous weights has been introduced recently. The mentioned results on the benefit of collaboration and computational complexity have been extended to capacities $\kappa > 1$ [4, 5].

Approximating the *maximum* travel distance of k agents has been studied for other tasks such as visiting a set of given arcs [15], or visiting all nodes of a tree [14]. Furthermore Demaine et al. [11] studied fixed-parameter tractability for minimizing both the *sum of* as well as the *maximum* travel distance of agents for several tasks such as pattern formation, parametrized by the number of agents k.

The problem of performing some collaborative task using *selfish agents* is well studied in algorithmic game theory and, in particular, in algorithmic *mechanism design* [19] where the system pays the agents in order to make sure that they report truthfully their costs. The existence of computationally feasible truthful mechanisms is one of the central questions, as truthfulness is often obtained by running an exact algorithm [20]. In addition, even for simple problems, like shortest path, the mechanism may have to pay a large amount of money to the agents [13, 2]. Network flow problems have been studied for the case when selfish agents own edges of the network and the algorithm designer must assign payments to the agents for the delivery of messages [1, 17].

2 Preliminaries

2.1 Mechanisms

A mechanism is a pair (A, P) where A is an algorithm and P a payment scheme which, for a given vector $w' = (w'_1, \dots, w'_n)$ of costs reported by the agents, computes a solution $A(w')$ and a payment $P_i(w')$ for each agent i.

Definition 1 (truthful mechanism)**.** *A mechanism (A, P) is* truthful *if truth-telling is a dominant strategy (utility maximizing) for all agents. That is, for any vector $w' = (w'_1, \dots, w'_n)$ of costs reported by the agents, for any i, and for any true cost w_i of agent i,*

$$P_i(w') - w_i \cdot d_i(A(w')) \leq P_i(w_i, w'_{-i}) - w_i \cdot d_i(A(w_i, w'_{-i}))$$

where $d_j(x)$ is the travel distance of agent j in solution x, $w'_{-i} := (w'_1, \dots, w'_{i-1}, w'_{i+1}, \dots, w'_n)$, and where $(w_i, w'_{-i}) := (w'_1, \dots, w'_{i-1}, w_i, w'_{i+1}, \dots, w'_n)$.

Truthfulness can be achieved through a well-known construction known as VCG mechanisms [25, 9, 16], which requires that the underlying algorithm satisfies certain ‘optimality’ conditions.

Definition 2 (VCG-based mechanism)**.** *A VCG-based mechanism is a pair (A, P) of the following form: For any vector $w' = (w'_1, \dots, w'_n)$ of costs reported by the agents, each agent i is payed an amount*

$$P_i(w') = Q_i(w'_{-i}) - \left(\sum\nolimits_{j \neq i} w'_j \cdot d_j(A(w'))\right), \tag{2}$$

where $Q_i()$ is a function independent of w'_i and $d_j(x)$ is the travel distance of agent j in solution x.

Intuitively speaking, these mechanisms turn out to be truthful, whenever the underlying algorithm minimizes the social cost with respect to a fixed subset of solutions (in particular, an optimal algorithm always yields a truthful mechanism).

Theorem 1 (Proposition 3.1 in [20]). *A VCG-based mechanism* (A, P) *is truthful if algorithm* A *minimizes the social cost over a fixed subset* R_A *of solutions. That is, there exists* R_A *such that, for every* w',

$$A(w') \in \arg\min_{x \in R_A}\{\textsc{cost}(x, w')\} \ .$$

The above result is a simple rewriting of the one in [20] which is originally stated for a more general setting, in which agent i valuates a solution x an amount $v_i(x)$, and $v_i()$ is the private information. Our setting is the special case in which these valuations are all of the form $v_i(x) = -w_i \cdot d_i(x)$ and w_i is the private information.

Definition 3 (voluntary participation). *A mechanism* (A, P) *satisfies the* voluntary participation *condition if truth-telling agents have always a nonnegative utility. That is, for every* $w' = (w'_1, \ldots, w'_n)$, *and for every agent* i,

$$P_i(w') - w'_i \cdot d_i(A(w')) \geq 0 \ .$$

We assume there are at least *two agents* (otherwise the problem is trivial and there is no point in doing mechanism design; also one can easily show that voluntary participation and truthfulness cannot be achieved in this case). Voluntary participation can be obtained by the standard Clarke pivot rule, that is, by setting in the payments (2) the functions $Q_i()$ as follows:

$$Q_i(w'_{-i}) := \textsc{cost}(A(\perp, w'_{-i}), w'_{-i}) \tag{3}$$

where $(\perp, w'_{-i})$ is the instance in which agent i is not present. Note that, if algorithm A runs in polynomial time, then also the payments can be computed in polynomial time: we only need to recompute n solutions using A and their costs.

The next is a well-known result, whose proof we give here for completeness.

Fact 1. *The VCG-based mechanism* (A, P) *with the payments in* (3) *satisfies voluntary participation if the algorithm satisfies the following condition. For any vector* w' *and for any agent* i,

$$\textsc{cost}(A(\perp, w'_{-i}), w') \geq \textsc{cost}(A(w'), w') \ . \tag{4}$$

Proof. Observe that the utility of agent i is

$$\begin{aligned} Q_i(w'_{-i}) - \left(\sum\nolimits_{j \neq i} w'_j \cdot d_j(A(w'))\right) - w_i \cdot d_i(A(w')) = \\ Q_i(w'_{-i}) - \textsc{cost}(A(w'), (w_i, w'_{-i})) = \\ \textsc{cost}(A(\perp, w'_{-i}), w'_{-i}) - \textsc{cost}(A(w'), (w_i, w'_{-i})) \ . \end{aligned}$$

When i is truth-telling we have $w'_i = w_i$, and $w' = (w_i, w'_{-i})$. Also, $\textsc{cost}(A(\perp, w_{-i}), w'_{-i}) = \textsc{cost}(A(\perp, w_{-i}), w')$ because i is not present in solution $A(\perp, w'_{-i})$. Hence the utility of i is $\textsc{cost}(A(\perp, w'_{-i}), w') - \textsc{cost}(A(w'), w')$, which is nonnegative because of (4). □

Definition 4 (Approximation mechanism). *A mechanism* (A, P) *is a* $c-$*approximation mechanism, if for every input vector of bids* w' *its algorithm* A *computes a solution* $A(w')$ *which is a* $c-$*approximation of a best solution, i.e.* $\textsc{cost}(A(w'), w') \leq c \cdot \textsc{cost}(\textsc{Opt}, w')$.

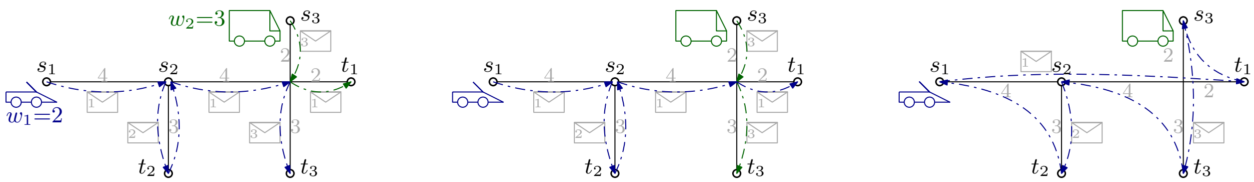


Figure 2: (left) Optimal solution of cost $2 \cdot 17 + 3 \cdot 4 = 46$ with collaboration on messages 1 and 3. (middle) Optimal solution among all non-collaborative solutions with energy cost $2 \cdot 16 + 3 \cdot 5 = 47$. (right) Optimal non-collaborative solution *with* both direct delivery and return to start; total energy cost $2 \cdot 2 \cdot 18 = 72$.

2.2 Collaboration of Agents

To describe a solution for an instance of the delivery problem we can (among other characteristics) elaborate on the following properties of the solution: How do agents work together on each message (*collaboration*), how are agents assigned to messages (*coordination*) and which route does each agent take (*planning*). Most of the mechanisms in this paper are based on the characterization of the *benefit of collaboration*:

Definition 5 (benefit of collaboration)**.** *Define R_{noC} as the set of solutions x in which there is no collaboration of the agents, meaning that each message is delivered by a single agent only. The ratio* $\min\limits_{x \in R_{\text{noC}}} \frac{\text{COST}(x,w)}{\text{COST}(\text{OPT},w)}$ *is called* benefit of collaboration (BoC).

Theorem 2 (Theorems 5 & 6 in [5])**.** *The benefit of collaboration is at most $1/\ln 2$ for a single message ($m = 1$), and at most 2 in general.*

The ratio of $1/\ln 2$ for a single message is tight, as can be seen in Figure 1. Furthermore, the general upper bound of 2 has been extended to any capacity $\kappa > 1$ and is tight as well ($\kappa, m \to \infty$) [4]. In the following, however, we make use of a BoC-characterization which guarantees additional properties:

Theorem 3 (Theorem 7 in [5])**.** *The benefit of collaboration is at most 2, even if we additionally restrict R_{noC} to include only solutions x that (i) transport every message directly from source to target without intermediate dropoffs, and (ii) return agents to their respective starting location.*

For an illustration of the relationship between optimal solutions, solutions in R_{noC} and solutions with the additional properties (i) and (ii), see Figure 2. From now on we restrict ourselves to solutions $x \in R_{\text{noC}}$ that satisfy the two additional conditions of Theorem 3.

Definition 6. *Define $R^{\star}_{\text{noC}}$ as the subset of solutions $x \in R_{\text{noC}}$ satisfying the two conditions (i) and (ii) of Theorem 3.*

3 Truthful Approximation Mechanisms

In this Section, we present several polynomial-time truthful mechanisms. Note that energy-efficient delivery is NP-hard to approximate to within any constant approximation ratio less than 367/366 [4, Theorem 9]. Hence, even in the best case, our goal can only be to guarantee a *constant-factor approximation* of the optimum in a truthful way.

Main algorithmic issue

In all problem versions we consider, we shall obtain truthful polynomial-time approximation mechanisms based on the following scheme:

1. Precompute in polynomial time a feasible subset R of solutions *independently of the input weights* w'; This set may depend on the name/index and on the position of the agents.
2. Among all precomputed solutions in R, return the best solution with respect to the input weights w', that is, a solution

$$x^* \in \arg\min_{x \in R}\{\textsc{cost}(x, w')\} \ .$$

Truthfulness then follows directly by Theorem 1. Note that, since we want polynomial running time, the first step selects a *polynomial* number of solutions (thus the second step is also polynomial). The main crux here is to make sure that, for all possible input weights w', the set R contains at least one *good approximation* (a solution $x \in R$ whose cost for w' is at most a constant factor above the optimum for w').

3.1 General setting

We start with the general setting of arbitrarily many messages m and arbitrarily many agents k. Our construction of a truthful approximation mechanism (A^*, P) for this setting relies on Theorem 1.

To this end, we define a fixed subset of solutions R_{A^*} and a polynomial-time algorithm A^*, such that for every vector of reported weights w', the algorithm A^* computes a solution S of optimum cost among all solutions in A^*, $S \in \arg\min_{x \in R_{A^*}} \textsc{cost}(x, w')$.

In a next step, we show that whenever the agents report truthfully ($w' = w$), the computed solution $x \in R_{A^*}$ has an energy cost that approximates the overall optimum energy cost by a constant factor of at most $4 \cdot \frac{w_{\max}}{w_{\min}}$, where $w_{\max} := \max_i w_i$ and $w_{\min} := \min_i w_i$.

A first approach We first analyze an existing approximation algorithm A_{pos} presented in [5], which computes a solution depending only on the position of the agent, but not on their declared costs. For an example of such a solution, see Figure 3 (top left) on the following page. As we will see, there is no mechanism (A_{pos}, P) which is both truthful *and* guarantees voluntary participation.

Theorem 4 (Theorem 13/18 in [4, 5]). *For any number of messages m, and any number of agents k, there exists a polynomial-time algorithm A_{pos} which computes a $(4 \cdot \frac{w_{\max}}{w_{\min}})$-approximation, for every w. The computed solution does not depend on the input weights w, but on the position of the agents only.*

Assume for the sake of contradiction there is a mechanism (A_{pos}, P) which is both truthful *and* guarantees voluntary participation: To guarantee voluntary participation we need the payments to satisfy $P_i(w') - w_i \cdot d_i(A_{pos}(w')) \geq 0$ for each agent i. Consider any agent i which is used in the solution, i.e. $d_i(A_{pos}(w_i, w'_{-i})) > 0$. Note that A_{pos} selected i independent of its weight w_i, hence d_i remains the same for all reported weights w'_i, i.e. we have $d_i := d_i(A_{pos}(w_i, w'_{-i})) = d_i(A_{pos}(w'_i, w'_{-i}))$. Let $P_i(w_i, w'_{-i})$ denote the payment to agent i when she reports her true weight w_i. Now consider a situation where agent i reports a different value $w'_i > P_i(w_i, w'_{-i})/d_i$ instead: To guarantee voluntary participation of agent i, the payment $P_i(w')$ needs to satisfy $P_i(w') - w'_i \cdot d_i \geq 0 \Leftrightarrow P_i(w') \geq w'_i \cdot d_i$, but $w'_i \cdot d_i > P_i(w_i, w'_{-i})$, contradicting the truthfulness of the mechanism (in other words: reporting an arbitrary high weight results in an arbitrary high payment).

Algorithm A^*

Input: Connected graph G, k agents, m messages, algorithm A_{pos} of Theorem 4 as a black box.
Output: A solution S with cost at most the cost of A_{pos}.

1: Compute the following $k+1$ solutions using A_{pos} as a black box subroutine:

$$x_0 := A_{pos}(w'), \text{ and } \forall i = 1, \ldots, k : x_{-i} := A_{pos}(\perp, w'_{-i}).$$

Let $R_{A^*} = \{x_0, x_{-1}, \ldots, x_{-k}\}$ be the subset of these solutions (feasible by connectivity of G).

2: Define algorithm A^* as taking the best among all these solutions with respect to the input weights w':

$$A^*(w') := \arg\min_{x \in R_{A^*}} \{\textsc{cost}(x, w')\} \ .$$

Refining the approach In order to obtain a truthful mechanism *with* voluntary participation we consider the algorithm A^* obtained from A_{pos} and a repeated application of algorithm A_{pos} on all subsets of $(k-1)$ agents. Since the underlying graph is connected and $k > 1$, running A_{pos} with only $k-1$ agents will always yield a feasible solution.

Theorem 5. *There exists a polynomial-time truthful VCG mechanism (A^*, P) satisfying voluntary participation and whose approximation ratio is at most the approximation ratio of A_{pos}.*

Proof. We use the VCG payments in (2) in the special case of (3) in order to satisfy also voluntary participation.

Since in Step 1 every solution is computed independently of the input weights, algorithm A^* satisfies the conditions of Theorem 1, which implies truthfulness. As for voluntary participation, we show that the algorithm also satisfies the condition in Fact 1 by Step 2 in A^*:

$$\textsc{cost}(A^*(w'), w') \overset{2.}{\leq} \textsc{cost}(A^*(\perp, w'_{-i}), w') \tag{5}$$

This is so because the solution $A^*(\perp, w'_{-i})$ is also a feasible solution that A^* considers on input w'.

Finally, the approximation guarantee is obvious since, by definition, A^* returns $A_{pos}(w)$ or a better solution. □

For an illustration of the computed set R_{A^*}, see Figure 3: The mechanism (A^*, P) will pick solution x_{-2} and award payments $P_1 = 46 - 10 = 36,\ P_2 = 0, P_3 = 160 - 10 = 150$.

3.2 Constant size settings

We now turn to developing truthful mechanisms where the approximation guarantee will be *an absolute constant*, therefore independent of the weights or the network distances (cf. Theorem 5).

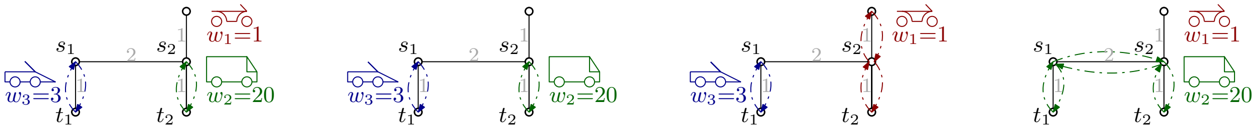


Figure 3: (from left to right) Original solution $x_0 = A_{pos}$ with energy cost $\textsc{cost}(x_0, w) = 46$, solution x_{-1} with $\textsc{cost}(x_{-1}, w) = 46$, solution x_{-2} with $\textsc{cost}(x_{-2}, w) = 10$ and solution x_{-3} with $\textsc{cost}(x_{-3}, w) = 160$.

To this end we provide approximation polynomial-time truthful mechanisms if either the number of messages m is constant or the number of agents k is constant – to the best of our knowledge the underlying algorithms are also the first absolute constant-factor approximations for the delivery problem.

We also show that by slightly deviating from the 2-step scheme given in the beginning of this section, the running time of the former algorithm (for small m) can be improved to $f(m) \cdot (kn)^{\mathcal{O}(1)}$, where f depends only on m and n is the number of nodes in the graph. Hence we get a truthful FPT-approximation mechanism, parametrized by the number of messages.

No collaboration Both mechanisms consider only solutions $x \in R^{\star}_{\text{noC}}$ where agents do not collaborate and follow the two properties given in Theorem 3. We are therefore left with the tasks of *coordinating* which agent gets assigned to which messages and *planning* in which order she delivers the assigned messages.

In other words, in every such a solution $x \in R^{\star}_{\text{noC}}$, each agent i is assigned a (possibly empty) block of messages $M_i(x) = \{i_1, i_2, \ldots, i_{|M_i(x)|}\}$. These message blocks are disjoint and form a partition of all m messages $\{1, \ldots, m\}$. Furthermore, agent i delivers each of its messages directly after picking it up at its source. These messages are processed in some order $\pi_x(i_1), \pi_x(i_2), \ldots, \pi_x(i_{|M_i(x)|})$, where π_x is a permutation of $\{1, \ldots, m\}$. Finally, i returns to its starting location p_i. Therefore, the travel distance $d_i(x)$ of agent i is

$$\begin{aligned} d_i(x) = \quad & \text{dist}\left(p_i, s_{\pi_x(i_1)}\right) + \sum_{j=1}^{|M_i|} \text{dist}\left(s_{\pi_x(i_j)}, t_{\pi_x(i_j)}\right) + \\ & \sum_{j=1}^{|M_i|-1} \text{dist}\left(t_{\pi_x(i_j)}, s_{\pi_x(i_{j+1})}\right) + \text{dist}\left(t_{\pi_x(i_{|M_i|})}, p_i\right), \end{aligned} \tag{6}$$

where $\text{dist}(u, v)$ denotes the distance of u and v in G.

Remark 1. *In the* $\text{dist}(u, v)$ *terms in* (6) *we are not interested in the actual route agent i takes, as long as it uses a shortest path between u and v.*

Sets and Lists To clarify the use of message blocks and message orders, we use the standard notion of sets and lists regarding partitions (see e.g. [6]): If we look at a partition of $\{1, \ldots, m\}$ into non-empty disjoint blocks, we can take into account the order of the elements within blocks, the order of the blocks, or both. We get four cases: sets of sets, sets of lists, lists of sets and lists of lists ([21, 22, 23, 24]).

However, in the delivery setting we can also have agents which are not used at all and therefore not assigned to any messages – a complete description (*coordination + planning*) of a solution $x \in R^{\star}_{\text{noC}}$ is therefore given by a *list of exactly k (possibly empty) lists*,

$$M = (M_1, \ldots, M_k)$$

where list M_i represents the sequence of the messages that agent i has to deliver in the order specified by M_i. Hence we immediately get a bijection from all lists of exactly k possibly empty lists to $R^{\star}_{\text{noC}}$ (modulo the equivalence between shortest paths – see Remark 1).

Fact 2. *For every such list of lists M, there is a solution $x_M \in R^{\star}_{\text{noC}}$ in which each agent i delivers the messages in M_i in the order specified by this list and the cost is minimized. Given M, such a solution can be computed in time poly(n, m, k).*

Algorithm A^m (for a constant number m of messages)

Input: Connected graph G, k agents, m messages.

Output: An optimal solution $S \in R^{\star}_{\mathrm{noC}}$.

1: Brute-force enumeration over all lists of exactly k possibly empty lists of the messages.

  foreach list of k lists **do**

    Add the corresponding solution (Fact 2) to the set of solutions R_{A^m}.

  end foreach

2: Define algorithm A^m as taking the best among all solutions in R_{A^m} with respect to the input weights w':

$$A^m(w') := \arg\min_{x \in R_{A^m}} \{\textsc{cost}(x, w')\} .$$

Constant number of messages m

We now look at the case of a constant number m of messages. By Theorem 3, the solution $S := \arg\min_{x \in R^{\star}_{\mathrm{noC}}} \textsc{cost}(x, w)$ is a 2−approximation of the optimum. First, we present an algorithm A^m which basically enumerates over all *lists of exactly k (possibly empty) lists* and adds the corresponding solution to a set R_{A^m} (where we get $R_{A^m} = R^{\star}_{\mathrm{noC}}$), as described in the first step of our scheme. Then, given an input vector of weights w', A^m chooses the best solution $A^m(w') \in \arg\min_{x \in R^{\star}_{\mathrm{noC}}} \{\textsc{cost}(x, w'\}$.

Theorem 6. *Algorithm A^m finds a best solution $S \in \arg\min_{x \in R^{\star}_{\mathrm{noC}}} \textsc{cost}(x, w)$ and can be implemented to run in time $\mathcal{O}\left(m!(k+m)^m \cdot poly(n, m, k)\right)$.*

Proof. Step 1 of A^m produces $\mathcal{O}(m! \cdot \binom{m+k-1}{k-1})$ many solutions and can be implemented in time $\mathcal{O}\left(m!(k+m)^m \cdot poly(m, k)\right)$ as follows: Since we look at a list of lists we have a total order on the messages. Hence we first enumerate in an outer loop over all $m!$ permutations, which can be done in time $\mathcal{O}(m!)$. Each such permutation also needs to be subdivided into exactly k possibly empty lists. There are $\binom{m+k-1}{k-1} \leq (k+m)^m$ many ways to do this (by placing $k-1$ delimiters at $m+k-1$ potential positions in time $\mathcal{O}\left(poly(m, k)\right)$).

Step 2 of A^m consists of computing the cost of all the solutions $x \in R^{\star}_{\mathrm{noC}}$, each of which can be done in time $\mathcal{O}\left(poly(n, k)\right)$. □

Theorem 7. *For a constant number of messages m, there exists a polynomial-time truthful VCG mechanism (A^m, P) satisfying voluntary participation and whose approximation ratio is at most 2.*

Proof. Theorem 6 says that the algorithm satisfies the conditions in Theorem 1, and thus truthfulness holds. Theorem 6 also implies the running time. The approximation is due to Theorem 3 and the definition of benefit of collaboration. The algorithm also satisfies the condition in Fact 1:

$$\textsc{cost}(A^m(w'), w') \leq \textsc{cost}(A^m(\perp, w'_{-i}), w') \tag{7}$$

Indeed, solution $y := A^m(\perp, w'_{-i})$ is also considered by the algorithm when all agents are present (input w'), because this solution corresponds to some list of k lists M in which agent i is not given any message (i.e. $M_i = \emptyset$). This list of lists is also considered on input w', and thus solution y is added to $R^{\star}_{\mathrm{noC}}$. □

We now show that the running time of the mechanism (A^m, P) can be improved. The main idea is to enumerate in A^m over all *sets of lists* instead of *list of lists* – i.e. during the enumeration we do not fix yet which agent gets which list of messages. Rather for each set of lists, we aim to

Algorithm A^m (improved version)

Input: Connected graph G, k agents, m messages.
Output: An optimal solution $S \in R^{\star}_{\text{noC}}$.

Enumerate over all *sets* of (non-empty) lists of the messages $1, \dots, m$.
foreach set of $\leq k$ lists **do**
  foreach Pair (agent i, list M_j) **do**
    1a: Assume agent i delivers the messages in M_j in their order.
    1b: Compute the cost $d_i(M_j)$ of doing so.
  end foreach
  2a: Build a complete bipartite graph Agents–Lists
    with edge costs $w_i \cdot d_i(M_j)$ for each edge $\{i, M_j\}$.
  2b: Find the best assignment Agents $\rightarrow$ Lists
    (by computing a maximum weighted bipartite matching).
  foreach Subset of $k-1$ agents **do**
    3: Repeat 2a, 2b for the subset of $k-1$ agents.
  end foreach
  4: Keep track of the best solution(s) found so far.
end foreach

directly compute an optimal assignment between the agents and the lists, thus deciding for every fixed set of lists in a *parallel way* the assigned list for every agent (instead of enumerating over all possible assignments as well). To do this we construct a complete bipartite graph between the agents and the lists, with edge weights indicating the energy an agent needs to deliver that list of messages (respecting the order of the list) and invoke a maximum weighted bipartite matching algorithm.

Furthermore we ensure that voluntary participation persists under the improved algorithm A^m, i.e. that the algorithm still computes the best solutions of the form $A(\perp, w'_{-i})$ for all i.

Theorem 8. *The running time of the truthful VCG mechanism* (A^m, P) *can be improved to a FPT, parametrized by the number of messages, of running time* $\mathcal{O}(f(m) \cdot poly(n,k))$*, where* $f(m) \in \mathcal{O}\left(e^{2\sqrt{m}-m} m^m \cdot poly(m)\right)$.

Proof. Consider first the running time of the improved algorithm A^m, which iterates over all sets of non-empty lists of the messages $1, \dots, m$. The number of sets of lists is known to be $\mathcal{O}(e^{2\sqrt{m}-m} m^m / poly(m))$ [22]. To this end we enumerate over all sets of lists without spending more than an additional $\mathcal{O}(poly(m))$-factor (over the number of sets of lists) on the running time. This can be done by enumerating over all *sets of sets* (by considering messages one-by-one and deciding whether to put them in a previously created subset or whether to start a new subset), followed by enumerating over all permutations of the messages inside each subset.

For each of the sets of at most k lists, we compute the best assignment of the k agents to the lists (say $l \leq k$ many) with a weighted bipartite matching as follows: On one side of the bipartite graph, we have k vertices, one for each agent. On the other side, we have one vertex per list. We take the complete bipartite graph between the two sides. This graph has at most $k \cdot l \leq k \cdot m$ edges. We compute the cost of an (agent, list)-edge as the energy cost of delivering all messages in the bundle in that order with that agent. This requires $\mathcal{O}(poly(n,m))$ time per edge and $\mathcal{O}(poly(n,k,m))$ time in total. A *maximum matching of minimum cost* in this graph gives the best assignment of agents to bundles and can be found in polynomial time, e.g. by using the Hungarian method [18] or the Successive shortest path algorithm [12].

It remains to show that truthfulness and voluntary participation also hold for the improved version of algorithm A^m. Truthfulness follows immediately from the fact that A^m still considers all solutions $x \in R^\star_{\text{noC}}$, albeit always several of them in a parallel fashion. To be able to apply Clarke's pivot rule to define the payments via $Q_i(w'_{-i})$ we make sure to also consider all solutions on all subsets of $k-1$ agents, see Step 3 of the improved algorithm A^m. □

Constant number of agents k

Next we look at settings with a constant number of agents k but an arbitrary number of messages m. In this case, even for $k = 1$ and independent of whether or not we restrict the messages to be transported from source to target without intermediate drop-offs, it is NP-hard to approximate $\min_x \textsc{cost}(x, w')$ to within any constant approximation ratio less than 367/366 [5, Theorem 9]. The bottleneck lies in finding the optimal permutation π_{OPT} to minimize the travel distances, see Equation (6). Therefore, for a fixed partition of the messages into subsets $M_i(x)$, we look for a message order $\pi_x(i_1), \pi_x(i_2), \ldots, \pi_x(i_{|M_i(x)|})$ given by a permutation π_x of $\{1, 2, \ldots, m\}$ such that we get an approximation guarantee on $d_i(x)$.

This can be modeled as the *stacker-crane problem*, which asks for the following: Given a weighted graph G_{SCP} with a set of *directed arcs* and a set of *undirected edges*, find the minimum tour that uses each arc at least once. Since we restrict ourselves to the two additional conditions (i) direct delivery of each message, (ii) return of each agent i in the end, we can model the transport of message j along a shortest path by a directed edge from s_j to t_j: Hence we choose the graph G_{SCP} to consist of all nodes $s_j, t_j,\ j = i_1, \ldots, i_{|M_i|}$, together with directed arcs (s_j, t_j) of weight $\text{dist}_G(s_j, t_j)$ (corresponding to the length of a shortest path between s_j and t_j in G) and undirected edges $\{t_{j1}, s_{j2}\}$ of weights $\text{dist}_G(t_{j1}, s_{j2})$ (corresponding to the original distance between t_{j1}, s_{j2}).

For the stacker-crane problem, a polynomial-time 1.8-approximation due to Frederickson et al. is known [14]. It remains to iterate over all assignments of the messages to the k agents as in the following Algorithm A^k.

Theorem 9. *Algorithm A^k finds an approximate solution S of* $\textsc{cost}(S, w') \leq 1.8 \cdot \min_{x \in R^\star_{\text{noC}}} \textsc{cost}(x, w')$ *in time $\mathcal{O}\left(poly(n, m, k) \cdot m^k\right)$.*

Algorithm A^k (for a constant number k of agents)

Input: Connected graph G, k agents, m messages.
Output: A 1.8-approximate solution $S \in R^\star_{\text{noC}}$.

1: Brute-force enumeration over all lists of exactly k possibly empty sets of the messages.
  foreach list of k sets $(M_1, \ldots, M_k)$ **do**
    foreach agent i **do**
      a: Model the delivery of the messages M_i (by agent i) as a Stacker-Crane problem.
      b: Compute a solution $x|_{M_i}$ such that $d_i(x)$ is a 1.8-approximation.
    end foreach
  c: Add the delivery problem solution x, combined from the k Stacker-Crane solutions $x|_{M_i}$, to the set of solutions R_{A^k}.
  end foreach

2: Define algorithm A^k as taking the best among all solutions in R_{A^k} with respect to the input weights w':

$$A^k(w') := \arg\min_{x \in R_{A^k}} \{\textsc{cost}(x, w')\} \ .$$

Proof. Algorithm A^k first enumerates over all lists of exactly k possibly empty sets of the messages. This can be implemented to run in time $\mathcal{O}(m^k)$ by choosing for each of the m messages the subset of messages M_i it belongs to.

We know that for each list of exactly k possibly empty lists there is a corresponding solution $x \in R^\star_{\text{noC}}$ and vice versa (Fact 2). In particular there exists a list $M = (M_1, \ldots, M_k)$ of lists $M_1, \ldots, M_k$ for each optimal solution $\textsc{Opt}(R^\star_{\text{noC}}) \in \arg\min_{x \in R^\star_{\text{noC}}} \{\textsc{cost}(x, w')\}$. Algorithm A^k at some point considers the list of exactly k possibly empty *sets* $M_1, \ldots, M_k$ (where there is no prescribed order of the elements in each of the M_i, $i = 1, \ldots, k$). Applying the stacker-crane approximation algorithm, A^k approximates each of the travelling distances $d_i(\textsc{Opt}(R^\star_{\text{noC}}))$ of the agents by a factor of at most 1.8. Hence A^k also considers a solution $S' \in R^\star_{\text{noC}}$ of cost

$$\begin{aligned}\textsc{cost}(S', w') = \sum_{i=1}^{k} d_i(S') \leq \sum_{i=1}^{k} 1.8 \cdot d_i(\textsc{Opt}(R^\star_{\text{noC}})) = 1.8 \cdot \sum_{i=1}^{k} d_i(\textsc{Opt}(R^\star_{\text{noC}}))\\ = 1.8 \cdot \min_{x \in R^\star_{\text{noC}}} \textsc{cost}(x, w').\end{aligned}$$

Since this solution x is contained in the built set R_{A^k}, we know that $S := A^k(w')$ has $\textsc{cost}(S, w') \leq \textsc{cost}(S', w') \leq 1.8 \cdot \min_{x \in R^\star_{\text{noC}}} \textsc{cost}(x, w')$. □

Theorem 10. *For a constant number of agents k, there exists a polynomial-time truthful VCG mechanism (A^k, P) satisfying voluntary participation whose approximation ratio is at most* 3.6*.*

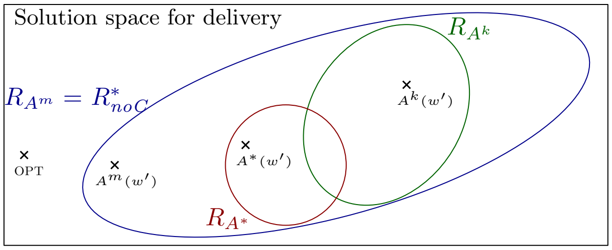
Proof. The approximation follows from Theorem 3 and Theorem 9. The latter theorem also implies the running time of the mechanism. Truthfulness and voluntary participation can be proved essentially in the same way as for Theorem 7. Indeed, the set of solutions R_{A^k} is computed independently of the input weights w', and thus the last step defining A^k satisfy the condition of Theorem 1 (implying truthfulness). As for voluntary participation, we observe that when A^k is run on input $(\perp, w'_{-i})$, the computed solution corresponds to some list of $k-1$ sets (agent i is not present), and the same solution is considered on input w' as a list of k sets, where one set is empty. This implies that the algorithm satisfies the condition in Fact 1:

$$\textsc{cost}(A^k(w'), w') \leq \textsc{cost}(A^k(\perp, w'_{-i}), w') \tag{8}$$

and thus voluntary participation. □

3.3 Comparison of the algorithms

We conclude our results on truthful approximation mechanisms with a comparison of the three given algorithms A^* (general setting), A^m (for a small number of messages m) and A^k (for a constant number of a agents k). Each of the three algorithms chooses an optimal solution from a respective set $R_{A^*}, R_{A^m}, R_{A^k}$. To ensure voluntary participation, in each set we include solutions where individual agents are not present – in this way we can set the payments according to Clarke's pivot rule (3). All three algorithms make use of Theorem 3 which bounds the Benefit of Collaboration $\textsc{BoC}$ for solutions in $R^\star_{\text{noC}}$ by 2 (this can be seen directly in the definition of the algorithms A^m, A^k, for algorithm A^* this follows from the black box algorithm A_{pos} [5, 4, Theorem 13/18]). In fact, the first version of the algorithm A^m computes a set R_{A^m} which consists of all the solutions in $R^\star_{\text{noC}}$ – the improved version of A^m discards solutions of high cost early on (while still considering solutions where individual agents are not present), in order to limit the set R_{A^m} to a small (FPT-) size, parametrized by the number of messages. Finally, we can compare the approximation guarantees.



Ratios between the computed solutions:

$\textsc{cost}(A^*(w')) \leq 4\frac{w_{\max}}{w_{\min}} \cdot \textsc{cost}(\textsc{Opt}, w')$ (Algorithm A_{pos})

$\textsc{cost}(A^m(w')) \leq 2 \cdot \textsc{cost}(\textsc{Opt}, w')$ (Benefit of Collaboration)

$\textsc{cost}(A^k(w')) \leq 1.8 \cdot \textsc{cost}(A^m(w'), w')$ (Stacker-Crane)

$\leq 3.6 \cdot \textsc{cost}(\textsc{Opt}, w')$ (Benefit of Collaboration)

Figure 4: Overview of the subsets of solutions R_{A^*}, R_{A^m}, R_{A^k} considered by the given algorithms and the approximation guarantees between the respective best solutions.

For algorithm A^* the approximation ratio of $4 \cdot \frac{w_{\max}}{w_{min}}$ follows from the (same) approximation ratio of A_{pos}. Both A^m and A^k have a factor of BoC ≤ 2, with an additional factor of 1.8 for A^k since we need to approximate the planning of each agent's tour (which we model with stacker-crane).

4 Single Message and Frugality

For the case of a *single message*, we define two truthful VCG mechanisms. The *optimal* mechanism which minimizes the social cost in Equation (1) using any number of agents, and the *lonely* mechanism which computes the solution of minimal cost under the constraint of using only one single agent. In both mechanisms, we use the VCG payments (2) with Clarke pivot rule (3) in order to satisfy voluntary participation.

Theorem 11 (Theorem 2 in [5])**.** *The optimal solution using a single agent, as well as the optimal solution using any number of agents, can be computed in polynomial time.*

Fact 3. *Both the exact and the lonely mechanisms are truthful since the algorithms are optimal with respect to a fixed subset of solutions, i.e., they satisfy the condition of Theorem 1. Moreover, the mechanisms run in polynomial time by Theorem 11.*

In the following we bound the total payments of these mechanisms compared with the cost of the optimal solution (for the given input). In other words, assuming the reported weights are the true weights ($w' = w$), we would like the mechanism to not pay much more than the optimum for these weights $w = w'$. This property is usually termed *frugality* [13, 2].

We first observe that, if we care more about the total payment made to the agents than about the optimality of the final solution, then in some instances it may pay off to run the lonely mechanism instead of the optimal mechanism. However, in other instances, the converse happens, meaning that neither mechanism is always better than the other.

Theorem 12. *For a single message, there are instances where the optimal mechanism pays a total amount of money larger than what the lonely mechanism does. Moreover, there are instances in which the opposite happens, that is, the lonely mechanism pays more than the optimal mechanism.*

Notation Throughout the proofs we use the following additional notation. For a generic algorithm A, we denote by $ALG = \textsc{cost}(A(w'), w')$ the cost of the solution computed by the algorithm on the reported costs w', and $ALG_{-i} = \textsc{cost}(A(\perp, w'_{-i}), w'_{-i})$ is the cost of an 'alternative solution' computed by the same algorithm on the instance $(\perp, w'_{-i})$ in which agent i is not present. Note that the VCG payments (2) with the Clarke pivot rule in (3) can be rewritten as

$$P_i(w') = ALG_{-i} - \big(ALG - w'_i \cdot d_i(A(w'))\big) \quad .$$

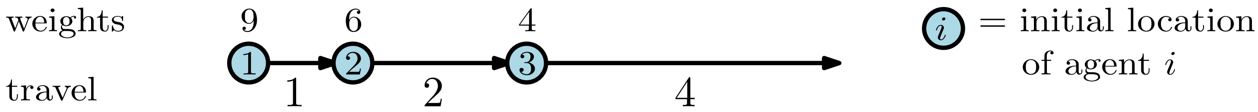


Figure 5: An example where the optimal mechanism pays in total more than what the lonely mechanism does. The message must travel from left to right along this path of length 7.

We further distinguish between OPT and $LOPT$, respectively the optimal and the lonely optimal algorithm's cost defined as above. In what follows, we will generally omit the agents' weights $w' = w$ from the formulas as these will be clear from the context.

Part 1 – The optimal mechanism pays more. The example in Figure 5 shows an instance and its optimal solution (we want the message to travel from left to right). The optimum for the instance in which agent 1 is not present has cost $OPT_{-1} = 40$ (let agent 3 do the whole work). More generally, one can check that $OPT_{-1} = OPT_{-2} = 40$ and $OPT_{-3} = 45$, and obviously $OPT = 9+12+16 = 37$. Plugging these values into the formula (2)-(3) for the payments, we get

$$P_1 = 40 - (37 - 9) = 12, \qquad P_2 = 40 - (37 - 12) = 15,$$
$$P_3 = 45 - (37 - 16) = 24,$$

for a total amount of money of 51 payed by this mechanism to the agents. The lonely mechanism will instead select agent 3 and pay only this agent an amount $LOPT_{-3} = 6 \cdot (2+2+4) = 48$, which is the lonely optimum for the instance where agent 3 is not present. □

Part 2 – The lonely mechanism pays more. The example in Figure 6 shows an instance and its optimal solution (we want the message to travel from left to right) whose cost is $OPT = 2-\epsilon+1+1 = 4-\epsilon$. The optimum for the instance in which agent 3 is not present has the same cost, that is, cost $OPT_{-3} = 4 - \epsilon$ (let agent 2 travel a distance 2 in total). Moreover, $OPT_{-1} = 4$ (agent 2 travels a distance 2) and $OPT_{-2} = 5 - 2\epsilon$ (agent 1 travels a distance 2 in total). Plugging these values into the formula (2)-(3) for the payments, we get

$$P_1 = 4 - 2 = 2, \qquad P_2 = (5 - 2\epsilon) - (3 - \epsilon) = 2 - \epsilon,$$
$$P_3 = (4 - \epsilon) - (3 - \epsilon) = 1,$$

for a total amount of money of $5-\epsilon$ payed by this mechanism to the agents. The lonely mechanism will instead select agent 2 and pay only this agent an amount $LOPT_{-2} = 1 \cdot 5 = 5$, which is the lonely optimum for the instance where agent 2 is not present. □

We remark that the payments given by (2)-(3) guarantee *voluntary participation*, and therefore the mechanism *must* pay a total amount of at least the optimum. We show below that both mechanisms are only a small constant factor away from this, except when a single agent can do the whole work for a much cheaper price than the others.

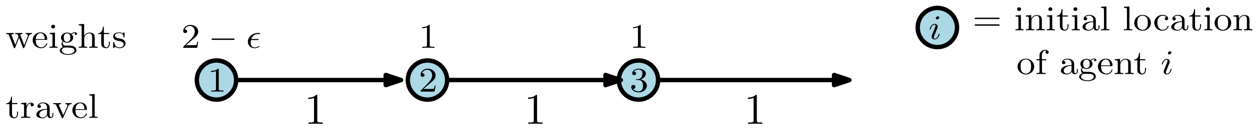


Figure 6: An example where the lonely mechanism pays more than what the optimal mechanism does. The message must travel from left to right along this path of length 3.

Definition 7 (monopoly free)**.** *We say that an instance with a single message is* monopoly free *if there is an optimal solution which uses at least two agents.*

The following example shows the need for this assumption.

Example 1. *Both the exact and the lonely mechanism perform equally bad if, for instance, there is only one very cheap agent and another very expensive one. Consider two agents of weights $w_1 = \epsilon$ small, and $w_2 = L$ large, both agent sitting on the starting position s of the message. In this case the two mechanisms output the same solution and the same payments, namely, agent 1 does the whole work and gets an amount of money given by the best alternative solution. That is, agent 1 gets a payment equal to $w_2 d = Ld$ where d is the length of the shortest s-t-path, while the optimum is $w_1 d = \epsilon d$.*

Theorem 13. *In any single message monopoly free instance, the optimal mechanism pays a total amount of money which is at most twice the optimum.*

Proof. The optimal solution selects a certain number $\ell \geq 2$ of agents and assigns to each of them some path. By renaming the agents, we can therefore assume that the optimum cost is of the form $OPT = w_1 d_1 + w_2 d_2 + \cdots + w_\ell d_\ell$, where no agent appears twice and the weights must satisfy $w_i \geq w_{i+1}$ and $w_i \leq 2w_{i+1}$, for otherwise agent i can replace agent $i+1$ or vice versa. We shall prove below that

$$OPT_{-i} \leq OPT + w_i d_i \tag{9}$$

and obtain from (2) that every agent is payed at most twice her cost, $P_i \leq OPT + d_i w_i - (OPT - d_i w_i) = 2 w_i d_i$, which then implies the theorem.

To complete the proof we show (9) by distinguishing two cases. For $i < \ell$ we can replace agent i with agent $i+1$ who then has to travel an additional distance of at most $2d_i$ to reach i and come back to its position. This gives an upper bound: $OPT_{-i} \leq OPT - w_i d_i + w_{i+1} 2 d_i \leq OPT + w_i d_i$, where the last inequality is due to $w_{i+1} \leq w_i$. For $i = \ell$ we replace agent i by agent $\ell - 1$ who travels an extra amount d_ℓ, and obtain this upper bound: $OPT_{-\ell} \leq OPT - w_\ell d_\ell + w_{\ell-1} d_\ell \leq OPT + w_\ell d_\ell$, where the last inequality is due to $w_{\ell-1} \leq 2 w_\ell$. □

Theorem 14. *In any single message monopoly free instance, the lonely mechanism pays at most $2BoC$ times the optimum, where $BoC = 1/\ln 2 \approx 1.44$ is the benefit of collaboration given by Theorem 2.*

Proof. This mechanism selects a *single* agent i for doing the whole work. We denote by $LOPT$ the optimum under this restriction, and by $LOPT_{-i}$ the same quantity for the instance in which agent i is removed. A crude upper bound on the payment can be obtained via Theorem 2, where the second inequality requires the instance to be monopoly free:

$$\begin{aligned} P_i = LOPT_{-i} \leq BoC \cdot OPT_{-i} &\overset{(9)}{\leq} BoC \cdot (OPT + w_i d_i) \\ &\leq BoC \cdot 2 \cdot OPT. \end{aligned}$$

□

Acknowledgments

We would like to thank Jérémie Chalopin, Shantanu Das, Yann Disser and Jan Hackfeld for some insightful discussions during this research. This work was partially supported by the SNF (Project 200021L_156620, Algorithm Design for Microrobots with Energy Constraints).

References

[1] R. Agarwal and Ö. Ergun. Mechanism design for a multicommodity flow game in service network alliances. *Operations Research Letters*, 36(5):520 – 524, 2008.

[2] A. Archer and É. Tardos. Frugal path mechanisms. In *13th ACM-SIAM Symposium on Discrete Algorithms SODA'02*, pages 991–999, 2002.

[3] 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 *23rd International Colloquium on Structural Information and Communication Complexity SIROCCO'16*, 2016.

[4] A. Bärtschi, J. Chalopin, S. Das, Y. Disser, D. Graf, J. Hackfeld, A. Labourel, and P. Penna. Energy-efficient Delivery by Heterogeneous Mobile Agents. *arXiv e-prints, CoRR*, abs/1610.02361, 2016.

[5] A. Bärtschi, J. Chalopin, S. Das, Y. Disser, D. Graf, J. Hackfeld, A. Labourel, and P. Penna. Energy-efficient Delivery by Heterogeneous Mobile Agents. In *34th International Symposium on Theoretical Aspects of Computer Science STACS'17*, 2017. to appear.

[6] D. Callan. Sets, Lists and Noncrossing Partitions. *Journal of Integer Sequences*, 11, Feb. 2008.

[7] J. Chalopin, S. Das, M. Mihalák, P. Penna, and P. Widmayer. Data delivery by energy-constrained mobile agents. In *9th International Symposium on Algorithms and Experiments for Sensor Systems, Wireless Networks and Distributed Robotics ALGOSENSORS'13*, pages 111–122, 2013.

[8] J. Chalopin, R. Jacob, M. Mihalák, and P. Widmayer. Data delivery by energy-constrained mobile agents on a line. In *41st International Colloquium on Automata, Languages, and Programming ICALP'14*, pages 423–434, 2014.

[9] E. H. Clarke. Multipart Pricing of Public Goods. *Public Choice*, pages 17–33, 1971.

[10] J. Czyzowicz, K. Diks, J. Moussi, and W. Rytter. Communication problems for mobile agents exchanging energy. In *23rd International Colloquium on Structural Information and Communication Complexity SIROCCO'16*, 2016.

[11] E. D. Demaine, M. Hajiaghayi, H. Mahini, A. S. Sayedi-Roshkhar, S. Oveisgharan, and M. Zadimoghaddam. Minimizing movement. *ACM Transactions on Algorithms (TALG)*, 5(3):1–30, 2009.

[12] J. Edmonds and R. M. Karp. Theoretical improvements in algorithmic efficiency for network flow problems. *Journal of the ACM (JACM)*, 19(2):248–264, 1972.

[13] E. Elkind, A. Sahai, and K. Steiglitz. Frugality in path auctions. In *Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 701–709. Society for Industrial and Applied Mathematics, 2004.

[14] P. Fraigniaud, L. Gasieniec, D. Kowalski, and A. Pelc. Collective tree exploration. In *6th Latin American Theoretical Informatics Symposium LATIN'04*, pages 141–151, 2004.

[15] G. N. Frederickson, M. S. Hecht, and C. E. Kim. Approximation algorithms for some routing problems. In *Proceedings of the 17th Annual Symposium on Foundations of Computer Science (FOCS)*, 1976.

[16] T. Groves. Incentive in Teams. *Econometrica*, 41:617–631, 1973.

[17] E. Kalai and E. Zemel. Generalized network problems yielding totally balanced games. *Operations Research*, 30(5):998–1008, 1982.

[18] H. W. Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

[19] N. Nisan and A. Ronen. Algorithmic mechanism design. *Games and Economic Behavior*, 35(1–2):166 – 196, 2001.

[20] N. Nisan and A. Ronen. Computationally feasible VCG mechanisms. *Journal of Artificial Intelligence Research (JAIR)*, 29:19–47, 2007.

[21] N. J. A. Sloane. A000110: Bell numbers: number of "sets of sets", Online Encyclopedia of Integer Sequences. `http://oeis.org/A000262`.

[22] N. J. A. Sloane. A000262: Number of "sets of lists", Online Encyclopedia of Integer Sequences. `http://oeis.org/A000262`.

[23] N. J. A. Sloane. A000670: Fubini numbers: number of "lists of sets", Online Encyclopedia of Integer Sequences. `http://oeis.org/A000262`.

[24] N. J. A. Sloane. A002866: Number of "lists of lists", Online Encyclopedia of Integer Sequences. `http://oeis.org/A000262`.

[25] W. Vickrey. Counterspeculation, Auctions and Competitive Sealed Tenders. *Journal of Finance*, pages 8–37, 1961.