Truthful Mechanisms for Delivery with Agents*

Andreas Bärtschi[1], Daniel Graf[2], and Paolo Penna[3]

1 **Department of Computer Science, ETH Zürich, Zürich, Switzerland**
andreas.baertschi@inf.ethz.ch
2 **Department of Computer Science, ETH Zürich, Zürich, Switzerland**
daniel.graf@inf.ethz.ch
3 **Department of Computer Science, ETH Zürich, Zürich, Switzerland**
paolo.penna@inf.ethz.ch

Abstract

We study the game-theoretic task of selecting mobile agents to deliver multiple items on a network. An instance is given by m packages (physical objects) which have to be transported between specified source-target pairs in an undirected graph, and k mobile heterogeneous agents, each being able to transport one package at a time. Following a recent model [6], each agent i has a different rate of energy consumption per unit distance traveled, i.e., its *weight*. 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 [6]. Finally, we show that for some instances in the case of a single package, the sum of the payments can be bounded in terms of the optimum.

1998 ACM Subject Classification F.2 Analysis of Algorithms and Problem Complexity

Keywords and phrases delivery, agent, energy optimization, approximation mechanism, frugality

Digital Object Identifier 10.4230/OASIcs.ATMOS.2017.2

1 Introduction

We study the *delivery* of physical objects (henceforth called *packages*) by mobile agents. Regardless of whether large volumes are transported by motor lorries or cargo airplanes, or whether autonomous drones deliver your groceries, the fuel or battery consumption of the transporting agent is a major cost factor. The main concern for algorithm-design thus focuses on an energy-efficient operation of these *agents*. The primary energy expense is 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 same entity which wants to deliver the packages [6]. It has been shown that there is a polynomial-time constant-factor approximation algorithm with the ratio depending on the energy consumption rates.

* This work was partially supported by the SNF (Project 200021L_156620, Algorithm Design for Microrobots with Energy Constraints).

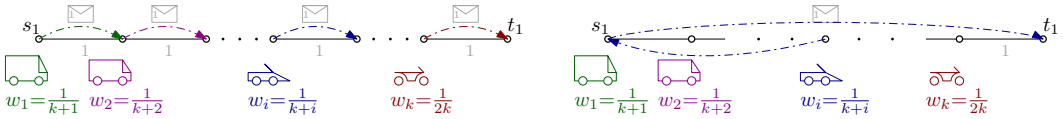


Figure 1 Delivery of a single package on a path of length k; consecutive agents of weight $w_i = \frac{1}{k+i}$. (left) Optimal solution using all agents; cost $= \sum_{i=1}^{k} w_i \cdot 1 = \sum_{i=1}^{k} \frac{1}{k+i} = \mathcal{H}_{2k} - \mathcal{H}_k \approx \ln 2$ $(k \to \infty)$. (right) Any 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 packages. Think for instance of a cargo company that hires subcontractors (like independent lorry drivers or individual couriers) to outsource the delivery of packages to these agents. In this scenario, the rate of energy consumption is a private value known only to the respective agent. Our goal is hence to design a mechanism for the cargo company to select agents, plan the agents' routes and reimburse them in a way such that:

1. the energy spent overall is as close to the optimum as possible,
2. each agent announces its true rate of energy consumption in its bid, and
3. each agent is reimbursed at least the cost of energy it needs to deliver all of its packages.

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 packages, 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 can carry at most one package at a time. An agent can pick up a package at some node u and drop the package again at some other node v. In that sense it is also possible to hand over a package j between agents if agent a drops it at a node where another agent b later picks up the same package, in which case we say that the agents *collaborate* on package j. A scenario in which each package 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, and 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): $\text{COST}(x, w) := \sum_{i=1}^{k} w_i \cdot d_i(x)$, where $w = (w_1, \ldots, w_k)$. We denote by OPT a solution with minimum cost among all solutions, $\text{OPT} \in \arg\min_x \text{COST}(x, w)$. 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 *private* piece of information known to agent i who can report a possibly different w_i'. 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 we are interested in the following:

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

- *Truthfulness.* For every agent, truth-telling is a *dominant strategy*, i.e., in no circumstance it is beneficial for an agent to misreport its cost. Thus, independently of the other agents, its utility (payment minus incurred cost) is maximized when reporting the true cost.
- *Frugality.* The total payment to the agents should be nearly optimal, that is, comparable to the total energy cost. Since agents should be payed at least their own cost when truthfully reporting (*voluntary participation*), the mechanism *must* pay at least $\textsc{cost}(x, w)$.

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 (w.r.t. the 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 [6]. 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 packages m is constant or the number of agents k is constant. For both cases we provide constant-factor approximation algorithms with approximation factors that are independent of the agents' weights. Both of these approximation algorithms satisfy sufficient conditions to be turned into truthful mechanisms with voluntary participation. The running time of the former algorithm can be improved to yield a *FPT*-approximation mechanism, parametrized by the number of packages; the latter runs in exponential time with k as the base of the exponential term.

Finally, in Section 4, we discuss the frugality of mechanisms for the case of a single package. 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 only. Although the latter results in a higher energy cost, it might need a smaller sum of 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).

Related work. *Energy-efficient* delivery has not been studied until recently. Most previous results are based on a model where the agents have uniform rates of energy-consumption but limited battery [8]. This restricts the possible movements of the agents – one gets the decision problem of whether the given packages can be delivered without exceeding the available battery levels. This turns out to be NP-hard even for a single package [9, 4] and even if energy can be exchanged between the agents [11]. The model of unbounded battery but heterogeneous weights has been introduced recently [6] (for the full version see [5]). Besides the mentioned approximation algorithm, it was shown that a restricted solution in which each package is delivered by a single agent approximates an optimum delivery (in which agents can handover a package to another agent) by a factor of at most 2. This result has been named the *Benefit of Collaboration*. Furthermore, the problem is NP-hard to approximate to within a small constant, even for a single agent.

Approximating the *maximum* travel distance of k agents has been studied for other tasks such as visiting a set of given arcs [17], or visiting all nodes of a tree [16]. Furthermore Demaine et al. [12] 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. These models consider only unweighted agents.

The problem of performing some collaborative task using *selfish agents* is well studied in algorithmic game theory and, in particular, in algorithmic *mechanism design* [21] where the system pays the agents in order to make sure that they report their costs truthfully. The existence of computationally feasible truthful mechanisms is one of the central questions, as truthfulness is often obtained by running an exact algorithm [22]. In addition, even for simple problems, like shortest path, the mechanism may have to pay a lot to the agents [14, 3]. Network flow problems have been studied for the case when selfish agents own edges of the network and the mechanism pays them in order to deliver packages [1, 19]. A setting where the transported packages are selfish entities which choose from fixed-route transportation providers was recently studied in [15]. In a sense, this is the reverse setting of our problem.

2 Preliminaries

Mechanisms. A mechanism is a pair (A, P) where A is an algorithm and P a payment scheme which, for a given vector $w' = (w'_1, \ldots, w'_k)$ 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 $w' = (w'_1, \ldots, w'_k)$ 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 $d_i(x)$ is the travel distance of agent i in solution x, and where $w'_{-i} := (w'_1, \ldots, w'_{i-1}, w'_{i+1}, \ldots, w'_k)$ and $(w_i, w'_{-i}) := (w'_1, \ldots, w'_{i-1}, w_i, w'_{i+1}, \ldots, w'_k)$.

Truthfulness can be achieved through a construction known as VCG mechanisms [27, 10, 18], 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 $w' = (w'_1, \ldots, w'_k)$ of costs reported by the agents, and for each agent i, there is a function $Q_i()$ independent of w'_i such that i is payed an amount of

$$P_i(w') = Q_i(w'_{-i}) - \left(\sum\nolimits_{j \neq i} w'_j \cdot d_j(A(w'))\right). \tag{1}$$

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 3** (Proposition 3.1 in [22])**.** *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', $A(w') \in \arg\min_{x \in R_A}\{\text{COST}(x, w')\}$.*

The above result is a simple rewriting of the one in [22] which is originally stated for a more general setting, in which agent i valuates a solution x by 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 (this setting is also called one-parameter [2]).

▶ **Definition 4** (Voluntary participation)**.** A mechanism (A, P) satisfies the *voluntary participation* condition if truth-telling agents have always a nonnegative utility. That is, for every $w' = (w'_1, \ldots, w'_k)$, 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, setting in the payments (1) the functions $Q_i()$ as follows:

$$Q_i(w'_{-i}) := \text{COST}(A(\perp, w'_{-i}), w'_{-i}) \tag{2}$$

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

▶ **Fact 5.** *The VCG-based mechanism (A, P) with the payments in (2) satisfies voluntary participation if the algorithm satisfies the following condition: For any vector w' and for any agent i, $\text{COST}(A(\perp, w'_{-i}), w') \geq \text{COST}(A(w'), w')$.*

Proof. Observe that the utility of agent i is

$$\begin{aligned} Q_i(w'_{-i}) - \Big(\sum_{j \neq i} w'_j \cdot d_j(A(w'))\Big) - w_i \cdot d_i(A(w')) &= Q_i(w'_{-i}) - \text{COST}(A(w'), (w_i, w'_{-i})) \\ &= \text{COST}(A(\perp, w'_{-i}), w'_{-i}) - \text{COST}(A(w'), (w_i, w'_{-i})). \end{aligned}$$

When i is truth-telling we have $w'_i = w_i$, and $w' = (w_i, w'_{-i})$. Also, $\text{COST}(A(\perp, w'_{-i}), w'_{-i}) = \text{COST}(A(\perp, w'_{-i}), w')$ because i is not present in solution $A(\perp, w'_{-i})$. Hence the utility of i is $\text{COST}(A(\perp, w'_{-i}), w') - \text{COST}(A(w'), w')$, which is non-negative by assumption. ◀

▶ **Definition 6** (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 $\text{OPT}(w')$, i.e., $\text{COST}(A(w'), w') \leq c \cdot \text{COST}(\text{OPT}(w'), w')$.

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 package (*collaboration*), how are agents assigned to packages (*coordination*) and which route does each agent take (*planning*). Most of the mechanisms in this paper are based on the characterizations of the *benefit of collaboration*:

▶ **Definition 7** (Benefit of collaboration)**.** Define R_{noC} as the set of solutions x in which there is no collaboration of the agents, meaning that each package is delivered by a single agent only. Define $R^{\star}_{\text{noC}} \subset R_{\text{noC}}$ as the subset of solutions $x \in R_{\text{noC}}$ in which

(i) every package is transported directly from source to target without intermediate dropoffs, and

(ii) every agent returns to its respective starting location.

The ratios $\text{BoC} := \min_{x \in R_{\text{noC}}} \frac{\text{COST}(x,w)}{\text{COST}(\text{OPT},w)}$ and $\text{BoC}^* := \min_{x \in R^{\star}_{\text{noC}}} \frac{\text{COST}(x,w)}{\text{COST}(\text{OPT},w)}$ are called *benefit of collaboration*.

▶ **Theorem 8** (Theorems 5, 6 & 7 in [6])**.** *For the the benefit of collaboration we have $\text{BoC} \leq 1/\ln 2$ for a single package ($m = 1$, for a tight lower bound example see Fig. 1), and $\text{BoC} \leq \text{BoC}^* \leq 2$ in general ($m \geq 1$, see for example Fig. 2).*

3 Truthful Approximation Mechanisms

In this section, we present two polynomial-time truthful mechanisms. Note that energy-efficient delivery is NP-hard to approximate to within any constant approximation ratio less

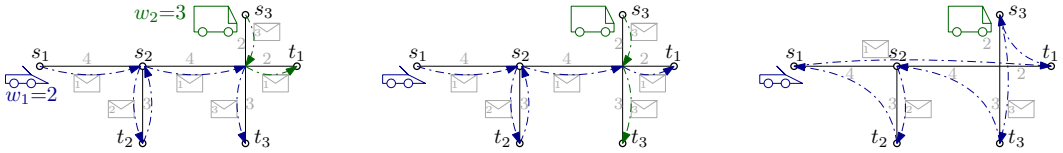


Figure 2 (left) Optimal solution of cost $2 \cdot 17 + 3 \cdot 4 = 46$ with collaboration on packages 1 and 3. (middle) Optimal solution among all non-collaborative solutions with energy cost $2 \cdot 16 + 3 \cdot 5 = 47$. (right) Optimal non-collaborative solution *with* direct delivery and return;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

than 367/366 [6, 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.

In the first part we present a polynomial-time truthful approximation mechanism with a constant-factor approximation guarantee *depending on the weights* of the agents. In the second part we turn to *absolute constant*-factor approximation, for which we require the number of packages m to be constant in order to get a polynomial-time mechanism. Similar techniques yield a mechanism running in exponential time for a constant number of agents k.

Main algorithmic issue. The two truthful polynomial-time approximation mechanisms we obtain are based on the following scheme (and the third approximation mechanism follows the same scheme but takes exponential time):

1. Precompute 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 $x^* \in \arg\min_{x \in R}\{\text{COST}(x, w')\}$.

Truthfulness then follows directly by Theorem 3. 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 A polynomial-time approximation mechanism

We start with the general setting of arbitrarily many packages m and arbitrarily many agents k. Our construction of a truthful approximation mechanism (A^*, P) for this setting relies on Theorem 3. 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 R_{A^*}, $S \in \arg\min_{x \in R_{A^*}} \text{COST}(x, w')$.

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 [6], which computes a solution depending only on the position of the agents, but not on their reported costs. Roughly speaking, A_{pos} connects the agent positions and the package sources and targets by a minimum spanning forest, subject to the following: In each tree, there is

(i) an edge between the corresponding source/target nodes s_j, t_j and
(ii) exactly one agent position p_i.

Algorithm A^*

Input: Connected graph G, k agents, m packages, black box algorithm A_{pos} (Thm. 9).
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:

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

All solutions in the set $R_{A^*} := \{x_0, x_{-1}, \ldots, x_{-k}\}$ are feasible by connectivity of G.

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

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

All edge lengths correspond to the distances in the original instance. Agent i then traverses its tree in a DFS-like fashion, crossing each edge twice and thus delivering all packages in the tree. For an example of such a solution, see Figure 3 (left).

▶ **Theorem 9** (Theorem 13 in [6]). *For any number of packages m and agents k, there exists a polynomial-time algorithm A_{pos} which computes a $(4 \cdot \frac{w_{\max}}{w_{\min}})$-approximation. The computed solution does not depend on the input weights w, only on the position of the agents.*

However, there is no mechanism (A_{pos}, P) which is both truthful *and* guarantees voluntary participation: Assume for the sake of contradiction there is such a mechanism. 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: For 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).

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 (for feasibility recall that G is connected and that $k > 1$).

▶ **Theorem 10.** *There exists a polynomial-time truthful VCG mechanism (A^*, P) satisfying voluntary participation with approximation ratio of at most the approximation ratio of A_{pos}.*

Proof. We use VCG payments (1) with (2) as $Q_i(w'_{-i}) := \textsc{cost}(A_{pos}(\perp, w'_{-i}), w'_{-i}) = \textsc{cost}(x_{-i}, w'_{-i})$. Since in Step 1 every solution is computed independently of the input weights, algorithm A^* satisfies the conditions of Theorem 3, which implies truthfulness. We show voluntary participation along the lines of Fact 5: First note that by Step 2 in A^*:

$$\textsc{cost}(A^*(w'), w') \leq \textsc{cost}(A_{pos}(\perp, w'_{-i}), w') \ , \tag{3}$$

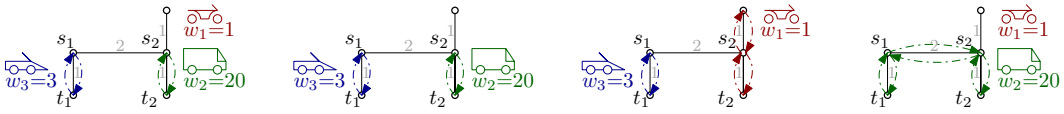


Figure 3 (from left to right) Original solution $x_0 = A_{pos}$ with energy cost $\text{COST}(x_0, w) = 46$, solutions x_{-1}, x_{-2}, x_{-3} with energy $\text{COST}(x_{-1}, w) = 46$, $\text{COST}(x_{-2}, w) = 10$ and $\text{COST}(x_{-3}, w) = 160$.

since the solution $x_{-i} = A_{pos}(\perp, w'_{-i})$ is a feasible solution that A^* considers on input w'. By the same argument, the approximation ratio of A^* is at most the approximation ratio of $A_{pos}(w)$. For voluntary participation, note that for $w'_i = w_i$, i's utility is

$$\begin{aligned} & Q_i(w'_{-i}) - \Big(\sum_{j \neq i} w'_j \cdot d_j(A^*(w'))\Big) - w_i \cdot d_i(A^*(w')) \\ &= Q_i(w'_{-i}) - \text{COST}(A^*(w'), (w_i, w'_{-i})) \\ &= \text{COST}(A_{pos}(\perp, w'_{-i}), w'_{-i}) - \text{COST}(A^*(w'), (w'_i, w'_{-i})) \overset{(3)}{\geq} 0. \end{aligned}$$ ◀

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 Absolute constant-factor approximation mechanisms

We now turn to developing truthful mechanisms where the approximation guarantee will be *an absolute constant*, therefore independent of the weights (cf. Theorems 9 and 10). To this end, we provide a polynomial-time truthful approximation mechanism if the number of packages m is constant. We also show that by slightly deviating from the 2-step scheme given in the beginning of this section, the running time of the algorithm can be improved to $f(m) \cdot (kn)^{\mathcal{O}(1)}$, where f depends only on m. Hence we get a truthful FPT-approximation mechanism, parametrized by the number of packages. To the best of our knowledge, the underlying algorithm is also the first absolute constant-factor approximation for the delivery problem. For the case of only constantly many agents k, we provide an exponential-time algorithm, where k is the base of the exponential term.

No collaboration. In the following, we consider only solutions $x \in R^{\star}_{\text{noC}}$ where agents do not collaborate and follow the two properties given in Theorem 8. We are therefore left with the tasks of *coordinating* which agent gets assigned to which packages and *planning* in which order she delivers the assigned packages. In other words, in every such solution $x \in R^{\star}_{\text{noC}}$, each agent i is assigned a (possibly empty) block of packages $M_i(x) = \{i_1, i_2, \ldots, i_{|M_i(x)|}\}$. These package blocks are disjoint and form a partition of all m packages $\{1, \ldots, m\}$. Furthermore, agent i delivers each of its packages directly after picking it up at its source. These packages 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, denoting the distance between u and v in G by $\text{dist}(u, v)$, the travel distance $d_i(x)$ of agent i can be written as

$$\begin{aligned} d_i(x) = \quad & \text{dist}\left(p_i, s_{\pi_x(i_1)}\right) + \sum_{j=1}^{|M_i|} \text{dist}\left(s_{\pi_x(i_j)}, t_{\pi_x(i_j)}\right) + \\ & \sum_{j=1}^{|M_i|-1} \text{dist}\left(t_{\pi_x(i_j)}, s_{\pi_x(i_{j+1})}\right) + \text{dist}\left(t_{\pi_x(i_{|M_i|})}, p_i\right). \end{aligned} \tag{4}$$

▶ Remark. In the $\text{dist}(u, v)$ terms in (4) we are not interested in the actual route agent i takes, as long as it uses a shortest path between u and v.

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

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

1: Brute-force enumeration over all lists of exactly k possibly empty lists of the packages.
foreach list of k lists **do**
Add the corresponding solution (Fact 11) 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':

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

Sets and lists. To clarify the use of package blocks and package orders, we use the standard notion of sets and lists regarding partitions (see e.g., [7]): 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 ([23, 24, 25, 26]). However, in the delivery setting we can also have agents which are not used at all and therefore not assigned to any packages – 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 packages 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 3.2).

▶ **Fact 11.** *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 packages in M_i in the order specified by this list and in which the cost is minimized. Given M, such a solution can be computed in time $poly(n, m, k)$.*

Constant number of packages m

We now look at the case of a constant number m of packages. By Theorem 8, the solution $S := \arg\min_{x \in R^\star_{\text{noC}}} \text{COST}(x, w)$ 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 $R_{A^m} = R^\star_{\text{noC}}$), as described in the first step of our scheme. Then, given an input vector of weights w', A^m chooses the best solution $A^m(w') \in \arg\min_{x \in R^\star_{\text{noC}}} \{\text{COST}(x, w'\}$.

▶ **Theorem 12.** *Algorithm A^m finds a best solution $S \in \arg\min_{x \in R^\star_{\text{noC}}} \text{COST}(x, w)$ 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 packages. 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 each solution $x \in R^\star_{\text{noC}}$ in time $\mathcal{O}\left(poly(n, m, k)\right)$. ◀

Algorithm A^m (improved version)

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

Enumerate over all *sets* of (non-empty) lists of the packages $1, \ldots, m$.
foreach set of $\leq k$ lists **do**
  foreach pair (agent i, list M_j) **do**
    1a: Assume agent i delivers the packages 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

▶ **Theorem 13.** *For a constant number of packages m, there exists a polynomial-time truthful VCG mechanism (A^m, P), satisfying voluntary participation, with approximation ratio ≤ 2.*

Proof. Theorem 12 says that the algorithm satisfies the conditions in Theorem 3, and thus truthfulness holds. Theorem 12 also implies the running time. The approximation is due to Theorem 8 and the definition of the benefit of collaboration BoC*. We next argue that the algorithm satisfies the condition in Fact 5, $\text{cost}(A^m(w'), w') \leq \text{cost}(A^m(\bot, w'_{-i}), w')$. Indeed, every solution $A^m(\bot, w'_{-i})$ is also considered by the algorithm when all agents are present (input w'), since this solution corresponds to some list of k lists M in which agent i is not given any package (i.e. $M_i = \emptyset$). Thus the solution $A^m(\bot, w'_{-i})$ is contained in $R^{\star}_{\text{noC}}$. ◀

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 packages. Rather for each set of lists, we aim to 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).

▶ **Theorem 14.** *The running time of the truthful VCG mechanism (A^m, P) can be improved to a FPT, parametrized by the number of packages, 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 packages $1, \ldots, m$. The number of sets of lists is known to be $\mathcal{O}(e^{2\sqrt{m}-m} m^m / poly(m))$ [24]. To this end we enumerate over all sets of lists while spending only 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 packages 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 packages 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 packages 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 [20] or the successive shortest path algorithm [13].

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 k different 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 packages m. In this case, even for $k = 1$ and independent of whether or not we restrict the packages to be transported from source to target without intermediate drop-offs, it is NP-hard to approximate $\min_x \text{COST}(x, w')$ to within any constant approximation ratio less than 367/366 [6, Theorem 9]. There, the bottleneck lies in finding the optimal permutation π_{OPT} to minimize the travel distances, see Equation (4). For $k > 1$, we first have to partition the packages into (possibly empty) subsets $M_1, \ldots, M_k$. Contrary to A^* and A^m, we will need exponential time $\mathcal{O}(k^m)$ to enumerate all partitions. Next, for every fixed partition of the packages into subsets M_i, we look for a package order $\pi_x(i_1), \pi_x(i_2), \ldots, \pi_x(i_{|M_i|})$ given by a permutation π_x of $\{1, 2, \ldots, m\}$ such that we get an approximation guarantee on $d_i(x)$.

Stacker-Crane problem. The latter 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 package, (ii) return of each agent i in the end, we can model the transport of package 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 node p_i and 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 $\{t_{j1}, s_{j2}\}, \{p_i, s_j\}, \{p_i, t_j\}$ of weights corresponding to the original distances $\text{dist}_G(t_{j1}, s_{j2})$ between t_{j1}, s_{j2} (respectively $\text{dist}_G(p_i, s_j)$, $\text{dist}_G(p_i, t_j)$). For the Stacker-Crane problem, a polynomial-time 1.8-approximation due to Frederickson et al. is known [16]. It remains to iterate over all assignments of the packages to the k agents as in the presented Algorithm A^k.

▶ **Theorem 15.** *Algorithm A^k runs in time $\mathcal{O}\left(poly(n, m, k) \cdot k^m\right)$ and finds an approximate solution S of $\text{COST}(S, w') \leq 1.8 \cdot \min_{x \in R^\star_{\text{noC}}} \text{COST}(x, w')$.*

Proof. Algorithm A^k first enumerates over all lists of exactly k possibly empty sets of the packages. This can be implemented to run in time $\mathcal{O}(k^m)$ by choosing for each of the m packages the subset of packages 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 11). In particular there exists a list $M = (M_1, \ldots, M_k)$ of lists $M_1, \ldots, M_k$ for each optimal solution

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

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

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 packages.
foreach list of k sets $(M_1, \ldots, M_k)$ **do**
foreach agent i **do**
a: Model the delivery of the packages 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 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':

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

$\text{OPT}(R^\star_{\text{noC}}) \in \arg\min_{x \in R^\star_{\text{noC}}} \{\text{COST}(x, w')\}$. 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 $d_i(\text{OPT}(R^\star_{\text{noC}}))$ 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

$$\text{COST}(S', w') = \sum_{i=1}^{k} w'_i \cdot d_i(S') \leq \sum_{i=1}^{k} w'_i \cdot 1.8 \cdot d_i(\text{OPT}(R^\star_{\text{noC}})) = 1.8 \cdot \min_{x \in R^\star_{\text{noC}}} \text{COST}(x, w').$$

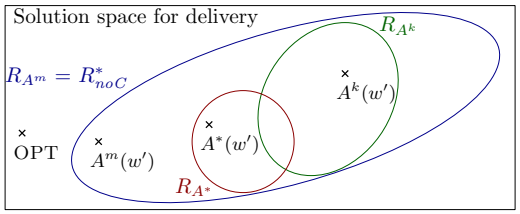
Since this solution S' is contained in the built set R_{A^k}, we know that $S := A^k(w')$ has $\text{COST}(S, w') \leq \text{COST}(S', w') \leq 1.8 \cdot \min_{x \in R^\star_{\text{noC}}} \text{COST}(x, w')$. ◀

▶ **Theorem 16.** *For a constant number of agents k, there exists an* exponential*-time truthful VCG mechanism (A^k, P), satisfying voluntary participation, with approximation ratio ≤ 3.6.*

Proof. The approximation follows from Theorem 8 and Theorem 15. 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 13. 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 3 (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 $\text{COST}(A^k(w'), w') \leq \text{COST}(A^k(\perp, w'_{-i}), w')$, i.e., Fact 5 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, polynomial time), A^m (FPT, parametrized by the number of packages m) and A^k (exponential time for a constant number of agents k), given in Figure 4. Each of the three algorithms chooses an optimal solution from a respective set



Ratios between the computed solutions:

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

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

$$\begin{aligned}\text{cost}(A^k(w')) &\leq 1.8 \cdot \text{cost}(A^m(w'), w') \quad \text{(Stacker-Crane)} \\ &\leq 3.6 \cdot \text{cost}(\text{Opt}, w') \quad \text{(Benefit of Collaboration)}\end{aligned}$$

Figure 4 Venn diagram of the subsets of solutions R_{A^*}, R_{A^m}, R_{A^k} considered by the given algorithms. Depending on the actual instance, the intersection of R_{A^*} and R_{A^k} might be empty.

$R_{A^*}, R_{A^m}, R_{A^k}$. 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 (2). All three algorithms make use of Theorem 8 which bounds the Benefit of Collaboration BoC* 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} [6, Theorem 13]). 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 packages. Finally, we can compare the approximation guarantees. 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 $\text{BoC}^* \leq 2$, 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 Package and Frugality

For the case of a *single package*, we define two truthful VCG mechanisms. The *optimal* mechanism which minimizes the social cost 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 (1) with Clarke pivot rule (2) in order to satisfy voluntary participation.

▶ **Theorem 17** (Theorem 2 in [6]). *For $m = 1$, 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 18.** *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 3. Moreover, the mechanisms run in polynomial time by Theorem 17.*

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* [14, 3].

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.

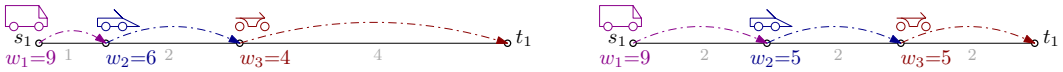


Figure 5 Delivery examples and their optimal solution. (left) The optimal mechanism pays more than the lonely mechanism. (right) The lonely mechanism pays more than the optimal mechanism.

In order to compare the optimal mechanism with the lonely mechanism, it is useful to define the following shorthands:

$$OPT = \text{COST}(A_{opt}(w'), w') \ , \qquad OPT_{-i} = \text{COST}(A_{opt}(\perp, w'_{-i}), w'_{-i}) \ ,$$
$$LOPT = \text{COST}(A_{lon}(w'), w') \ , \qquad LOPT_{-i} = \text{COST}(A_{lon}(\perp, w'_{-i}), w'_{-i}) \ ,$$

where A_{opt} is the optimal algorithm, and A_{lon} is the lonely algorithm, that is, the one computing the optimal solution using a single agent. Then the VCG payments (1) with the Clarke pivot rule in (2) can be rewritten as follows in the two cases:

$$P_i(w') = OPT_{-i} - (OPT - w'_i \cdot d_i(A_{opt}(w'))) \ , \tag{5}$$
$$P_i(w') = LOPT_{-i} - (LOPT - w'_i \cdot d_i(A_{lon}(w'))) \ . \tag{6}$$

Below, we usually omit the agents' weights $w' = w$ whenever they are clear from the context.

▶ **Fact 19.** *The lonely mechanism pays the selected agent i an amount $LOPT_{-i}$, and the other agents get no payment. This is because $LOPT = w'_i \cdot d_i(A_{lon}(w'))$ when i is selected, and for all non-selected agent $j \neq i$, $LOPT = LOPT_{-j}$ and $d_j(A_{lon(w')}) = 0$.*

▶ **Theorem 20.** *For a single package, 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.*

Proof. First we prove that there are instances, in which the optimal mechanism pays more. The example in Figure 5 (left) shows an instance and its optimal solution. 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 $OPT = 9 + 12 + 16 = 37$. Using these values for the payments (5), we get $P_1 = 40 - (37 - 9) = 12$, $P_2 = 40 - (37 - 12) = 15$, $P_3 = 45 - (37 - 16) = 24$, for a total amount of 51 payed by this mechanism to the agents. The lonely mechanism will instead select agent 3 and pay only this agent (see Fact 19) an amount $LOPT_{-3} = 6 \cdot (2+2+4) = 48$, which is the lonely optimum for the instance where agent 3 is not present.

On the other hand, the example in Figure 5 (right) shows an instance in which the lonely mechanism pays more. The optimal solution has cost is $OPT = 18 + 10 + 10 = 38$ and we get $OPT_{-1} = 40$, $OPT_{-2} = 46$ and $OPT_{-3} = 38$, resulting in the payments $P_1 = 40 - (38 - 18) = 20$, $P_2 = 46 - (38 - 10) = 18$, $P_3 = 38 - (38 - 10) = 10$ for a total amount of 48. The lonely mechanism will instead select agent 2 and pay $LOPT_{-2} = 50$. ◀

We remark that the payments given by (1) and (2) guarantee *voluntary participation*, and therefore the mechanism *must* pay a total amount of at least the optimum. We show that both mechanisms pay only a small constant factor over the optimum, except when a single agent can do the work for a much cheaper price than the others (as shown in Example 22):

▶ **Definition 21** (monopoly-free)**.** We say that an instance with a single package is *monopoly-free* if there is an optimal solution which uses at least two agents.

▶ **Example 22.** 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 agents sitting on the starting position s_1 of the package. In this case the two mechanisms output the same solution and payment: Agent 1 does the whole work and gets an amount of money given by the best alternative solution. We have $P_1 = w_2 \operatorname{dist}(s_1, t_1) = L \operatorname{dist}(s_1, t_1)$, while the optimum is $w_1 \operatorname{dist}(s_1, t_1) = \epsilon \operatorname{dist}(s_1, t_1)$.

▶ **Theorem 23.** *In any single package 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

$$OPT_{-i} \leq OPT + w_i d_i. \tag{7}$$

Using VCG payments (1), we obtain from (7) that every agent is payed at most twice her cost, $P_i \leq OPT + w_i d_i - (OPT - w_i d_i) = 2w_i d_i$, which then implies the theorem. To complete the proof, we show (7) 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: $OPT_{-i} \leq OPT - w_i d_i + w_{i+1} 2d_i \leq OPT + w_i d_i$, 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 $w_{\ell-1} \leq 2w_\ell$. ◀

▶ **Theorem 24.** *In any single package monopoly-free instance, the lonely mechanism pays at most $2BoC$ times the optimum, where $BoC = 1/\ln 2 \approx 1.44$ (by Theorem 8).*

Proof. Let i be the selected agent. A crude upper bound on the sum of the payments can be obtained via Fact 19, where the second inequality requires the instance to be monopoly-free:
$P_i = LOPT_{-i} \leq \mathrm{BoC} \cdot OPT_{-i} \overset{(7)}{\leq} \mathrm{BoC} \cdot (OPT + w_i d_i) \leq \mathrm{BoC} \cdot 2 \cdot OPT.$ ◀

5 Conclusion and open questions

This work initiates the study of truthful mechanisms for delivery in a natural setting where mobile agents are *selfish* and can speculate about their own energy consumption rate. We considered mechanisms which are *truthful*, run in *polynomial-time*, have a worst-case *approximation* guarantee and, possibly, do not pay the agents abnormal amounts (*frugality*). We provide such polynomial-time truthful approximation mechanisms for two cases:

1. when the consumption rate of different agents are similar, i.e., the ratio $\max \frac{w_i}{w_j}$ is bounded (see Theorem 10) and
2. when the number of packages m is small (see Theorem 14).

For a single package, we also gave bounds on the frugality of two natural truthful mechanisms which return the optimum or an approximation of the optimum.

The main open question is whether there exists a polynomial-time constant-factor approximation independent of the weights. This remains an intriguing open question even for a constant number of agents k, for which we presented an exponential-time truthful approximation mechanism.

Acknowledgments. We would like to thank Jérémie Chalopin, Shantanu Das, Yann Disser, Jan Hackfeld, and Peter Widmayer for some insightful discussions. We are very grateful for the feedback provided by the anonymous reviewers.

References

1 Richa Agarwal and Özlem Ergun. Mechanism design for a multicommodity flow game in service network alliances. *Operations Research Letters*, 36(5):520–524, 2008. doi:10.1016/j.orl.2008.04.007.

2 Aaron Archer and Éva Tardos. Truthful mechanisms for one-parameter agents. In *42nd IEEE Symposium on Foundations of Computer Science FOCS'01*, pages 482–491, 2001.

3 Aaron Archer and Éva Tardos. Frugal path mechanisms. In *13th ACM-SIAM Symposium on Discrete Algorithms SODA'02*, pages 991–999, 2002.

4 Andreas Bärtschi, Jérémie Chalopin, Shantanu Das, Yann Disser, Barbara Geissmann, Daniel Graf, Arnaud Labourel, and Matús Mihalák. Collaborative Delivery with Energy-Constrained Mobile Robots. In *23rd International Colloquium on Structural Information and Communication Complexity SIROCCO'16*, 2016.

5 Andreas Bärtschi, Jérémie Chalopin, Shantanu Das, Yann Disser, Daniel Graf, Jan Hackfeld, Arnaud Labourel, and Paolo Penna. Energy-efficient Delivery by Heterogeneous Mobile Agents. *arXiv e-prints, CoRR*, abs/1610.02361, 2016. URL: http://arxiv.org/abs/1610.02361, arXiv:1610.023610.

6 Andreas Bärtschi, Jérémie Chalopin, Shantanu Das, Yann Disser, Daniel Graf, Jan Hackfeld, Arnaud Labourel, and Paolo Penna. Energy-efficient Delivery by Heterogeneous Mobile Agents. In *34th Symposium on Theoretical Aspects of Computer Science (STACS 2017)*, volume 66 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 10:1–10:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPIcs.STACS.2017.10.

7 David Callan. Sets, Lists and Noncrossing Partitions. *Journal of Integer Sequences*, 11, February 2008. arXiv:0711.4841.

8 Jérémie Chalopin, Shantanu Das, Matús Mihalák, Paolo Penna, and Peter 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. doi:10.1007/978-3-642-45346-5_9.

9 Jérémie Chalopin, Riko Jacob, Matús Mihalák, and Peter 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. doi:10.1007/978-3-662-43951-7_36.

10 Edward H. Clarke. Multipart Pricing of Public Goods. *Public Choice*, pages 17–33, 1971.

11 Jurek Czyzowicz, Krzysztof Diks, Jean Moussi, and Wojciech Rytter. Communication problems for mobile agents exchanging energy. In *23rd International Colloquium on Structural Information and Communication Complexity SIROCCO'16*, 2016.

12 Erik D. Demaine, Mohammadtaghi Hajiaghayi, Hamid Mahini, Amin S. Sayedi-Roshkhar, Shayan Oveisgharan, and Morteza Zadimoghaddam. Minimizing movement. *ACM Transactions on Algorithms (TALG)*, 5(3):1–30, 2009. doi:10.1145/1541885.1541891.

13 Jack Edmonds and Richard M. Karp. Theoretical improvements in algorithmic efficiency for network flow problems. *Journal of the ACM (JACM)*, 19(2):248–264, 1972.

14 Edith Elkind, Amit Sahai, and Ken Steiglitz. Frugality in path auctions. In *15th ACM-SIAM Symposium on Discrete Algorithms SODA'04*, pages 701–709, 2004.

15 Dimitris Fotakis, Laurent Gourvès, and Jérôme Monnot. Selfish transportation games. In *43rd International Conference on Current Trends in Theory and Practice of Computer SOFSEM'17*, pages 176–187, 2017.

16 Pierre Fraigniaud, Leszek Gasieniec, Dariusz R. Kowalski, and Andrzej Pelc. Collective tree exploration. In *6th Latin American Theoretical Informatics Symposium LATIN'04*, pages 141–151, 2004.

17 Greg N. Frederickson, Matthew S. Hecht, and Chul E. Kim. Approximation algorithms for some routing problems. In *17th IEEE Symposium on Foundations of Computer Science FOCS'76*, 1976.

18 Theodore Groves. Incentive in Teams. *Econometrica*, 41:617–631, 1973.

19 Ehud Kalai and Eitan Zemel. Generalized network problems yielding totally balanced games. *Operations Research*, 30(5):998–1008, 1982. doi:10.1287/opre.30.5.998.

20 Harold W. Kuhn. The Hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

21 Noam Nisan and Amir Ronen. Algorithmic mechanism design. *Games and Economic Behavior*, 35(1–2):166–196, 2001. doi:10.1006/game.1999.0790.

22 Noam Nisan and Amir Ronen. Computationally feasible VCG mechanisms. *Journal of Artificial Intelligence Research (JAIR)*, 29:19–47, 2007.

23 Neil James Alexander Sloane. A000110: Bell numbers: number of "sets of sets", Online Encyclopedia of Integer Sequences. http://oeis.org/A000262.

24 Neil James Alexander Sloane. A000262: Number of "sets of lists", Online Encyclopedia of Integer Sequences. http://oeis.org/A000262.

25 Neil James Alexander Sloane. A000670: Fubini numbers: number of "lists of sets", Online Encyclopedia of Integer Sequences. http://oeis.org/A000262.

26 Neil James Alexander Sloane. A002866: Number of "lists of lists", Online Encyclopedia of Integer Sequences. http://oeis.org/A000262.

27 William Vickrey. Counterspeculation, Auctions and Competitive Sealed Tenders. *Journal of Finance*, pages 8–37, 1961.