Energy-Efficient Delivery by Heterogeneous Mobile Agents*†

Andreas Bärtschi[1], Jérémie Chalopin[2], Shantanu Das[3], Yann Disser[4], Daniel Graf[5], Jan Hackfeld[6], and Paolo Penna[7]

1 **Department of Computer Science, ETH Zürich, Zürich, Switzerland**
baertschi@inf.ethz.ch
2 **LIF, CNRS and Aix-Marseille Université, Marsaille, France**
jeremie.chalopin@lif.univ-mrs.fr
3 **LIF, CNRS and Aix-Marseille Université, Marsaille, France**
shantanu.das@lif.univ-mrs.fr
4 **TU Darmstadt, Germany**
disser@mathematik.tu-darmstadt.de
5 **Department of Computer Science, ETH Zürich, Zürich, Switzerland**
daniel.graf@inf.ethz.ch
6 **Institut für Mathematik, TU Berlin, Berlin, Germany**
hackfeld@math.tu-berlin.de
7 **Department of Computer Science, ETH Zürich, Zürich, Switzerland**
paolo.penna@inf.ethz.ch

Abstract

We consider the problem of delivering m messages between specified source-target pairs in an undirected graph, by k mobile agents initially located at distinct nodes of the graph. Each edge has a designated length and 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. Unlike previous related work, we consider heterogeneous agents with different rates of energy consumption (weights w_i). To solve the delivery problem, agents face three major challenges: *Collaboration* (how to work together on each message), *Planning* (which route to take) and *Coordination* (how to assign agents to messages).

We first show that the delivery problem can be 2-approximated *without* collaborating and that this is best possible, i.e., we show that the *benefit of collaboration* is 2 in general. We also show that the benefit of collaboration for a single message is $1/\ln 2 \approx 1.44$. Planning turns out to be NP-hard to approximate even for a single agent, but can be 2-approximated in polynomial time if agents have unit capacities and do not collaborate. We further show that coordination is NP-hard even for agents with unit capacity, but can be efficiently solved exactly if they additionally have uniform weights. Finally, we give a polynomial-time $(4 \max \frac{w_i}{w_j})$-approximation for message delivery with unit capacities.

1998 ACM Subject Classification F.2 Analysis of Algorithms and Problem Complexity, F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases message delivery, mobile agents, energy optimization, approximation algorithms

Digital Object Identifier 10.4230/LIPIcs.STACS.2017.10

* A full version of the paper is available at https://arxiv.org/abs/1610.02361

† This work was partially supported by the project ANR-ANCOR (anr-14-CE36-0002-01), the SNF (project 200021L_156620) and the DFG Priority Programme 1736 "Algorithms for Big Data" (grant SK 58/10-1).

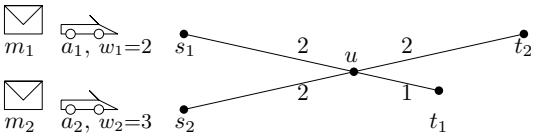
1 Introduction

Recent technological progress in robotics allows the mass production of inexpensive mobile robots which can be used to perform a variety of tasks autonomously without the need for human intervention. This gives rise to a variety of algorithmic problems for teams of autonomous robots, hereafter called *mobile agents*. We consider here the delivery problem of moving some objects or messages between various locations. A mobile agent corresponds to an automated vehicle that can pick up a message at its source and deliver it to the intended destination. In doing so, the agent consumes energy proportional to the distance it travels. Our goal is to design a centralized algorithm for the agents such that the total energy consumed is minimized.

In general the agents may not be all identical; some may be more energy efficient than others if they use different technologies or different sources of power. We assume each agent has a given *weight* which is the rate of energy consumption per unit distance traveled by this agent (here we use the term *weights*, since the rates are weights of the objective function). Moreover, the agents may start from distinct locations. Thus it may be sometimes efficient for an agent to carry the message to some intermediate location and hand it over to another agent which carries it further towards the destination. On the other hand, an agent may carry several messages at the same time. Finding an optimal solution that minimizes the total energy cost involves scheduling the moves of the agents and the points where they pick up or handover the messages. We study this problem (called WeightedDelivery) for a graph G which connects all sources and destinations. The objective is to deliver m messages between specific source-target pairs using k agents located in arbitrary nodes of G. Note that this problem is distinct from the connectivity problems on graphs or network flow problems since the initial location of the agents are in general different from the sources where the messages are located, which means we need to consider the cost of moving the agents to the sources in addition to the cost of moving the messages. Furthermore, there is no one-to-one correspondence between the agents and the messages in our problem.

Previous approaches to energy-efficient delivery of messages by agents have focused on a bottleneck where the agents have limited energy (battery power) which restricts their movements [1, 8]. The decision problem of whether a single message can be delivered without exceeding the available energy for any agent is known as the DataDelivery problem [9] or the BudgetedDelivery problem [4] and it was shown to be weakly NP-hard on paths [9] and strongly NP-hard on planar graphs [4].

Our Model. We consider an undirected graph $G = (V, E)$. Each edge $e \in E$ has a *cost* (or *length*) denoted by l_e. The length of a simple path is the sum of the lengths of its edges. The distance between nodes u and v is denoted by $d_G(u, v)$ and is equal to the length of the shortest path from u to v in G. There are k mobile agents denoted by $a_1, \ldots a_k$ and having weights $w_1, \ldots w_k$. These agents are initially located on arbitrary nodes $p_1, \ldots, p_k$ of G. We denote by $d(a_i, v)$ the distance from the initial location of a_i to node v. Each agent can move along the edges of the graph. Each time an agent a_i traverses an edge e it incurs an energy cost of $w_i \cdot l_e$. Furthermore there are m pairs of (source, target) nodes in G such that for $1 \leq i \leq m$, a message has to be delivered from source node s_i to a target node t_i. A message can be picked up by an agent from any node that it visits and it can be carried to any other node of G, and dropped there. The agents are given a *capacity* κ which limits the number of messages an agent may carry simultaneously. There are no restrictions on how much an agent may travel. We denote by d_j the total distance traveled by the j-th



An optimal, feasible schedule:
$(a_1, s_1, m_1, +), (a_2, s_2, m_2, +), (a_1, u\ , m_1, -),$
$(a_2, u\ , m_2, -), (a_1, u\ , m_2, +), (a_1, t_2, m_2, -),$
$(a_2, u\ , m_1, +), (a_2, t_1, m_1, -)$

Figure 1 Example of an optimal, feasible schedule for two messages and two agents.

agent. WeightedDelivery is the optimization problem of minimizing the total energy $\sum_{j=1}^{k} w_j \cdot d_j$ needed to deliver all messages.

A *schedule* S describes the actions of all agents as a sequence (ordered list) of pick-up actions $(a_j, p, m_i, +)$ and drop-off actions $(a_j, q, m_i, -)$, where each such tuple denotes the action of agent a_j moving from its current location to node p (node q) where it picks up message m_i (drops message m_i, respectively). A schedule S implicitly encodes all the pick-up and drop-off times and it is easy to compute its total energy use of $\text{cost}(S) := \sum_{j=1}^{k} w_j d_j$. We denote by $S|_{a_j}$ the subsequence of all actions carried out by agent a_j and by $S|_{m_i}$ the subsequence of all actions involving pick-ups or drop-offs of message m_i. We call a schedule *feasible* if every pick-up action $(\_, p, m_i, +)$, $p \neq s_i$, is directly preceded by a drop-off action $(\_, p, m_i, -)$ in $S|_{m_i}$ and if all the messages get delivered, see Figure 1.

Our Contribution. Solving WeightedDelivery naturally involves simultaneously solving three subtasks, *collaboration*, *individual planning*, and *coordination*: First of all, if multiple agents work on the same message, they need to collaborate, i.e., we have to find all intermediate drop-off and pick-up locations of the message. Secondly, if an agent works on more than one message, we have to plan in which order it wants to approach its subset of messages. Finally, we have to coordinate which agent works on which subset of all messages (if they do this without collaboration, the subsets form a partition, otherwise the subsets are not necessarily pairwise disjoint). Even though these three subtasks are interleaved, we investigate collaboration, planning and coordination separately in the next three sections. This leads us to a polynomial-time approximation algorithm for WeightedDelivery, given in Section 5.

In Section 2 we consider the *Collaboration* aspect of WeightedDelivery. We first present a polynomial time solution for WeightedDelivery when there is only a single message ($m = 1$). The algorithm has complexity $O(|V|^3)$ irrespective of the number of agents k. In general, we show that any algorithm that only uses one agent for delivering every message cannot achieve an approximation ratio better than what we call the *benefit of collaboration* (BoC) which is at least $1/\ln\left((1 + 1/(2m))^m (1 + 1/(2m + 1))\right)$. We show this to be tight for $m = 1$ (where $BoC \geq 1/\ln 2$) and $m \to \infty$ (where $BoC \to 2$).

In Section 3 we look at the *Planning* aspect of WeightedDelivery. Individual planning by itself turns out to be NP-hard on planar graphs and NP-hard to approximate within a factor of less than $\frac{367}{366}$. On the positive side, we give approximation guarantees for restricted versions of WeightedDelivery which turn out to be useful for the analysis in Section 5.

In Section 4 we study the *Coordination* aspect of WeightedDelivery. Even if collaboration and planning are taken care of (i.e., a schedule is fixed except for the assignment of agents to messages), Coordination also turns out to be NP-hard even on planar graphs. The result holds for any capacity, including $\kappa = 1$. This setting, however, becomes tractable if restricted to uniform weights of the agents.

In Section 5 we give a polynomial-time approximation algorithm for WeightedDelivery with an approximation ratio of $4 \cdot \max \frac{w_i}{w_j}$ for $\kappa = 1$. Due to the limited space, some proofs are omitted, but can be found in the full version of the paper [6].

Related Work. The problem of communicating or transporting goods between sources and destinations in a graph has been well studied in a variety of models with different optimization criteria. The problem of finding the smallest subgraph or tree that connects multiple sources and targets in a graph is called the *point-to-point connection problem* and is known to be NP-hard [25]. The problem is related to the more well-known generalized Steiner tree problem [28] which is also NP-hard. Unlike these problems, the maximum flow problem in a network [14], puts a limit on the number of messages that can be transported over an edge, which makes the problem easier allowing for polynomial time solutions. In all these problems, however, there are no agents carrying the messages as in our problem.

For the case of a single agent moving in a graph, the task of optimally visiting all nodes, called the *Traveling salesman problem* or visiting all edges, called the *Chinese postman problem* have been studied before. The former is known to be NP-hard [2] while the latter can be solved in $O(|V|^2|E|)$ time [13]. For metric graphs, the traveling salesman problem has a polynomial-time $\frac{3}{2}$-approximation for tours [10] and for paths with one fixed endpoint [18]. For multiple identical agents in a graph, Demaine et al. [12] studied the problem of moving the agents to form desired configurations (e.g. connected or independent configurations) and they provided approximation algorithms and inapproximability results. Bilo et al. [7] studied similar problems on visibility graphs of simple polygons and showed many motion planning problems to be hard to approximate.

Another optimization criteria is to minimize the maximum energy consumption by any agent, which requires partitioning the given task among the agents. Frederickson et al. [17] studied this for uniform weights and called it the *k-stacker-crane problem* and they gave approximation algorithms for a single agent and multiple agents. Also in this minmax context, the problem of visiting all the nodes of a tree using k agents starting from a single location is known to be NP-hard [16]. Anaya et al. [1] studied the model of agents having limited energy budgets. They presented hardness results (on trees) and approximation algorithms (on arbitrary graphs) for the problem of transferring information from one agent to all others (*Broadcast*) and from all agents to one agent (*Convergecast*). For the same model, message delivery between a single *s-t* node pair was studied by Chalopin et al. [8, 9, 4] as mentioned above. A recent paper [11] shows that these three problems remain NP-hard for general graphs even if the agents are allowed to exchange energy when they meet.

2 Collaboration

In this section, we examine the *collaboration* of agents: Given for each message m_i all the agents $a_{i1}, a_{i2}, \ldots, a_{ix}$ which at some point carry the message, we need to find all pick-up and drop-off locations (handovers) $h_1, \ldots, h_y$ for the schedule entries $(a_{i1}, \_, m_i, +)$, $(a_{i1}, \_, m_i, -)$, $\ldots$, $(a_{ix}, \_, m_i, -)$. Note, that in general we can have more than two action quadruples $(a_{ij}, \_, m_i, +/-)$ per agent a_{ij}. When there is only a single message overall ($m = 1$), we will use a structural result to tie together WeightedDelivery and Collaboration. For multiple messages, however, this no longer holds: In this case, we analyze the benefit we lose if we forgo collaboration and deliver each message with a single agent.

2.1 An Algorithm for WeightedDelivery of a Single Message

▶ **Lemma 1.** *In any optimal solution to* WeightedDelivery *for a single message, if the message is delivered by agents with weights $w_1, w_2, \ldots w_k$, in this order, then*

(i) *$w_i \geq w_j$ whenever $i < j$, and*

(ii) *without loss of generality, $w_i \neq w_j$ for $i \neq j$.*

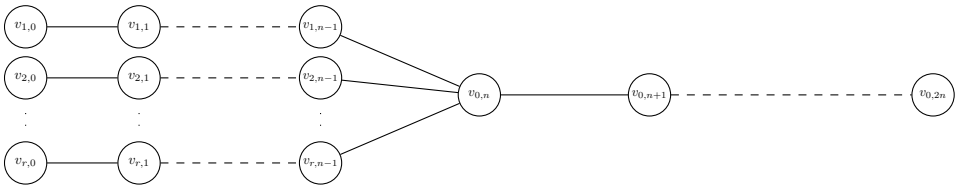


Figure 2 Lower bound construction for the benefit of collaboration.

Hence there is an optimal schedule S in which no agent a_j has more than one pair of pick-up/drop-off actions.

▶ **Theorem 2.** *An optimal solution of* WeightedDelivery *of a single message in a graph $G = (V, E)$ with $k \leq |V|$ agents can be found in $O(|V|^3)$ time.*

Proof. We use the properties of Lemma 1 to create an auxiliary graph on which we run Dijkstra's algorithm for computing a shortest path from s to t. Given an original instance of single-message WeightedDelivery consisting of the graph $G = (V, E)$, with $s, t \in V$, we obtain the auxiliary, *directed* graph $G' = (V', E')$ as follows:

- For each node $v \in V$ and each agent a_i, there is a node v_{a_i} in G'.
  Furthermore G' contains two additional vertices s and t.
- For $1 \leq i \leq k$, there is an arc (s, s_{a_i}) of cost $w_i \cdot d_G(p_i, s)$ and an arc (t_{a_i}, t) of cost 0.
- For $(u, v) \in E$ and $1 \leq i \leq k$, there are two arcs (u_{a_i}, v_{a_i}) and (v_{a_i}, u_{a_i}) of cost $w_i \cdot l_{(u,v)}$.
- For $u \in V$ and agents a_i, a_j with $w_i > w_j$, there is an arc (u_{a_i}, u_{a_j}) of cost $w_j \cdot d_G(p_j, u)$.

Note that any solution to the WeightedDelivery that satisfies the properties of Lemma 1 corresponds to some s-t-path in G' such that the cost of the solution is equal to the length of this path in G' and vice versa. This implies that the length of the shortest s-t path in G' is the cost of the optimal solution for WeightedDelivery in G. Assuming that $k \leq |V|$, the graph G' has $|V| \cdot k + 2 \in O(|V|^2)$ vertices and at most $2k + (k^2|V| + |V|^2 k)/2 - |V| \cdot k \in O(|V|^3)$ arcs. The edge-costs of the graph G' can be computed in $O(|V|^3)$ time if we use the *Floyd Warshall* all pair shortest paths algorithm [15, 27] in G. Finally, we compute the shortest path from s to t in G' in time $O(|V|^3)$, using Dijkstra's algorithm with Fibonacci heaps. ◀

Unfortunately, the structural properties of Lemma 1 do not extend to multiple messages. In the next two subsections we investigate how the quality of an optimal solution changes if we only allow every message to be transported by one agent. Different messages may still be transported by different agents and one agent may also transport multiple messages at the same time as long as the number of messages is at most the capacity κ. To this end we define the *Benefit of Collaboration* as the cost ratio between an optimal schedule Opt and a best-possible schedule without collaboration S, $\text{BoC} = \min_S \text{cost}(S)/\text{cost}(\text{Opt})$.

2.2 Lower Bound on the Benefit of Collaboration

▶ **Theorem 3.** *On instances of* WeightedDelivery *with agent capacity κ and m messages, an algorithm using one agent for delivering every message cannot achieve an approximation ratio better than $1/\ln\left((1 + 1/(2r))^r (1 + 1/(2r + 1))\right)$, where $r := \min\{\kappa, m\}$.*

Proof. Consider the graph $G = (V, E)$ given in Figure 2, where the length l_e of every edge e is $1/n$. This means that G is a star graph with center $v_{0,n}$ and $r + 1$ paths of total length 1 each. We have r messages and message i needs to be transported from $v_{i,0}$ to $v_{0,2n}$ for

$i = 1, \dots, r$. There further is an agent $a_{i,j}$ with weight $w_{i,j} = \frac{2r}{2r+j/n}$ starting at every vertex $v_{i,j}$ for $(i,j) \in \{1,\dots,r\} \times \{0,\dots,n-1\} \cup \{0\} \times \{n,\dots,2n\}$.

We first show the following: If any agent transports s messages $i_1, \dots, i_s$ from $v_{i_j,0}$ to $v_{0,2n}$, then this costs at least $2s$. Note that this implies that any schedule S for delivering all messages by the agents such that every message is only carried by one agent satisfies $\text{COST}(S) \geq 2r$.

So let an agent $a_{i,j}$ transport s messages from the source to the destination $v_{0,2n}$. Without loss of generality let these messages be $1, \dots, s$, which are picked up in this order. By construction, agent $a_{i,j}$ needs to travel a distance of at least $\frac{j}{n}$ to reach message 1, then distance 1 to move back to $v_{0,n}$, then distance 2 for picking up message i and going back to $v_{0,n}$ for $i = 2, \dots, s$, and finally it needs to move distance 1 from $v_{0,n}$ to $v_{0,2n}$. Overall, agent $a_{i,j}$ therefore travels a distance of at least $2s + \frac{j}{n}$. The overall cost for agent $a_{i,j}$ to deliver the s messages therefore is at least $(2s + \frac{j}{n}) \cdot w_{i,j} = (2s + \frac{j}{n}) \cdot \frac{2r}{2r+j/n} \geq (2s + \frac{j}{n}) \cdot \frac{2s}{2s+j/n} = 2s$.

Now, consider a schedule S_{col}, where the agents collaborate, i.e., agent $a_{i,j}$ transports message i from $v_{i,j}$ to $v_{i,j+1}$ for $i = 1, \dots, r$, $j = 0, \dots, n-1$, where we identify $v_{i,n}$ with $v_{0,n}$. Then agent $a_{0,j}$ transports all r messages from $v_{0,j}$ to $v_{0,j+1}$ for $j = n, \dots, 2n-1$. This is possible because $r \leq \kappa$ by the choice of r. The total cost of this schedule is given by

$$\text{COST}(S_{\text{col}}) = r \cdot \int_0^1 f_{\text{step}}(x)dx + \int_1^2 f_{\text{step}}(x)dx,$$

where $f_{\text{step}}(x)$ is a step-function defined on $[0,2]$ giving the current cost of transporting the message, i.e., $f_{\text{step}}(x) = \frac{2r}{2r+j/n}$ on the interval $[j/n, (j+1)/n)$ for $j = 0, \dots, 2n-1$. The first integral corresponds to the first part of the schedule, where the r messages are transported separately and therefore the cost of transporting message i from $v_{i,j}$ to $v_{i,j+1}$ is exactly $\int_{j/n}^{(j+1)/n} f_{\text{step}}(x)dx = \frac{1}{n} \cdot \frac{2r}{2r+j/n}$. The second part of the schedule corresponds to the part, where all r messages are transported together by one agent at a time.
Observe that the function $f(x) = 2r \cdot \frac{1}{2r-1/n+x}$ satisfies $f(x) \geq f_{\text{step}}(x)$ on $[0,2]$, hence

$$\begin{aligned}\text{COST}(S_{\text{col}}) &\leq r \int_0^1 f(x)dx + \int_1^2 f(x)dx = 2r \left(r \ln(2r - \tfrac{1}{n} + x)\Big|_0^1 + \ln(2r - \tfrac{1}{n} + x)\Big|_1^2 \right) \\ &= 2r \ln \left(\left(\tfrac{2r-1/n+1}{2r-1/n} \right)^r \left(\tfrac{2r-1/n+2}{2r-1/n+1} \right) \right) \overset{n\to\infty}{\to} 2r \ln \left(\left(1 + \tfrac{1}{2r}\right)^r \left(1 + \tfrac{1}{2r+1}\right) \right).\end{aligned}$$

Thus, the approximation ratio of an algorithm transporting every message by only one agent is bounded from below by $BoC \geq \min_S \frac{\text{COST}(S)}{\text{COST}(S_{\text{col}})} \geq 1/\ln \left(\left(1 + \frac{1}{2r}\right)^r \left(1 + \frac{1}{2r+1}\right) \right)$. ◀

By observing that $\lim_{r\to\infty} 1/\ln\left((1 + 1/(2r))^r (1 + 1/(2r+1)) \right) = 1/\ln\left(e^{1/2}\right) = 2$, we obtain the following corollary.

▶ **Corollary 4.** *A schedule for* WeightedDelivery *where every message is delivered by a single agent cannot achieve an approximation ratio better than* 2 *in general, and better than* $1/\ln 2 \approx 1.44$ *for a single message.*

2.3 Upper Bounds on the Benefit of Collaboration

We now give tight upper bounds for Corollary 4. The following theorem shows that the benefit of collaboration is 2 in general. We remark that finding an optimal schedule in which every message is transported from its source to its destination by one agent, is already NP-hard, as shown in Theorem 8.

▶ **Theorem 5.** *Let* Opt *be an optimal schedule for a given instance of* WeightedDelivery. *Then there exists a schedule S such that every message is only transported by one agent and* $\text{cost}(S) \leq 2 \cdot \text{cost}(\text{Opt})$.

Proof Sketch. We may assume (without loss of generality) that the optimal schedule Opt transports each message along a simple path, and we construct a directed multigraph G_S with the same set of vertices and an arc for every move of an agent in Opt. We label every arc by the exact set of messages that were carried during the corresponding move in Opt. For every arc in G_S we add a backwards arc with the same label.

Obviously, every connected component of G_S is Eulerian, and we claim that any agent in each component can follow some Eulerian tour that allows to deliver all messages. In particular, the agent needs exactly twice as many moves as the total number of moves of all agents in the component in Opt. If we choose the cheapest agent (in terms of weight) in each component, we obtain a tour with at most twice the cost of Opt.

We compute the Eulerian for the cheapest agent of a component as a combination of multiple tours, respecting arc labels in the following sense: During every move along a forward arc, the agent carries the exact set of messages prescribed by the arc label, and during every move along a backward arc, the agent does not carry any messages. This ensures that all messages travel along the same path as in Opt. Whenever the agent is at a vertex v and is missing message i in order to proceed along some path, this means the current vertex must lie on i's path in Opt, and thus there must be a path of backwards edges to the current location of i. The agent follows this path and recursively brings the message back to v. In the process, more recursive calls may be necessary, but we can prove that there cannot be a circular dependence between messages.

Therefore, the procedure eventually terminates after computing a closed tour. Note that, so far, the tour is still "virtual" in the sense that the agent didn't actually move but merely computed the tour. We remove the tour from G_S, update all message positions, and recursively apply the procedure starting from the last vertex along the tour that is still adjacent to untraversed edges. By combining all (virtual) tours that we obtain in the recursion, we eventually get a Eulerian tour for the agent that obeys all arc labels. This means that the agent can successfully simulate all moves in Opt while ensuring that it is carrying the required messages before each move. ◀

Single Message. For the case of a single message, we can improve the upper bound of 2 on the benefit of collaboration from Theorem 5, to a tight bound of $1/\ln 2 \approx 1.44$.

The idea of the proof is to use that the weights are non-increasing by Lemma 1. After scaling appropriately, we assume the message path to be the interval $[0, 1]$ and then choose a b such that the function $\frac{b}{x+1}$ is a lower bound on the weight of the agent transporting the message at point x on the message path. The intersection of $\frac{b}{x+1}$ and the step-function f representing the weight of the agent currently transporting the message then gives an agent, which can transport the message with at most $(1/\ln 2)$-times the cost of an optimal schedule.

▶ **Theorem 6.** *For* WeightedDelivery *with $m = 1$, there exists a $(1/\ln 2)$-approximation algorithm using a single agent.*

No Intermediate Dropoffs. For the capacities $\kappa = 1$ and $\kappa = \infty$, the upper bound of 2 on the benefit of collaboration still holds if we additionally demand that each message is carried by its single agent without any intermediate dropoffs. We will make use of this result later in the approximation algorithm for WeightedDelivery with $\kappa = 1$ (Section 5).

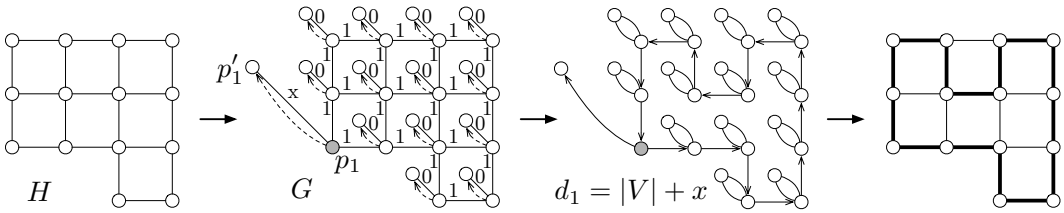


Figure 3 Finding a Hamiltonian cycle via WeightedDelivery with a single agent. Picking x to be large enough, e.g. $x = |V|$, allows us to enforce that the agent will end in p_1'.

▶ **Theorem 7.** *Let* Opt *be an optimal schedule for a given instance of* WeightedDelivery *with* $\kappa \in \{1, \infty\}$. *Then there exists a schedule* S *such that* (i) *every message is only transported by a single agent, with exactly one pick-up and one drop-off,* (ii) $\text{cost}(S) \leq 2 \cdot \text{cost}(\text{Opt})$, *and* (iii) *every agent* a_j *returns to its starting location* p_j.

3 Planning

We now look in isolation at the problem of ordering the messages within the schedule of an agent, which we call *Planning*. Formally, the *Planning* aspect of WeightedDelivery is the following task: Given a schedule S and one of its agents a_j, reorder the actions in $S|_{a_j}$ in such a way that the schedule remains feasible and the costs are minimized.

Generally speaking, for a complex schedule with many message handovers, the reordering options for a single agent a_j might be very limited. First of all, we must respect the capacity of a_j, i.e., in every prefix of $S|_{a_j}$, the number of pick-up actions $(a_j, *, *, +)$ cannot exceed the number of drop-off actions $(a_j, *, *, -)$ by more than κ. Even then, reordering $S|_{a_j}$ might render S infeasible because of conflicts with some other subschedule $S|_{a_x}$. But *Planning* also includes the instances where a single agent delivers all the messages, one after the other straight to the target, and where the only thing that has to be decided is the ordering. We show now that in this setting, where there is no non-trivial *coordination* or *collaboration* aspect, WeightedDelivery is already NP-hard.

▶ **Theorem 8.** Planning *of* WeightedDelivery *problem is* NP*-hard for all capacities* κ *even for a single agent on a planar graph.*

The hardness follows by a reduction from Hamiltonian cycles on a grid graph H, a problem shown to be NP-hard by Itai et al. [21]: We put an isolated message at every node of H, forcing the agent to visit each node exactly once. A longer edge for the isolated message at the start forces the agent to come back to the start node towards the end, see Figure 3.

Using similar ideas, we can use recent results for the approximation hardness of metric TSP [22] to immediately show that *Planning* of WeightedDelivery can not be approximated arbitrarily well, unless P = NP.

▶ **Theorem 9.** *It is* NP*-hard to approximate the* Planning *of* WeightedDelivery *to within any constant approximation ratio less than 367/366.*

3.1 Polynomial-time Approximation for Planning in Restricted Settings

Motivated by Theorem 7, we now look at the restricted setting of planning for a feasible schedule S^R of which we know that each message is completely transported by some agent a_j without intermediate drop-offs, i.e., for every message m_i there must be an agent j with

$S^R|_{m_i} = (a_j, s_i, m_i, +), (a_j, t_i, m_i, -)$. This allows us to give polynomial-time approximations for planning with capacity $\kappa \in \{1, \infty\}$:

▶ **Theorem 10.** *Let S^R be a feasible schedule for a given instance of* WeightedDelivery *with the restriction that $\forall i\ \exists j:\ S^R|_{m_i} = (a_j, s_i, m_i, +), (a_j, t_i, m_i, -)$. Denote by $\text{Opt}(S^R)$ a reordering of S^R with optimal cost. There is a polynomial-time planning algorithm* Alg *which gives a reordering $\text{Alg}(S^R)$ such that $\text{cost}(\text{Alg}(S^R)) \leq 2 \cdot \text{cost}(\text{Opt}(S^R))$ if $\kappa = 1$ and $\text{cost}(\text{Alg}(S^R)) \leq 3.5 \cdot \text{cost}(\text{Opt}(S^R))$ if $\kappa = \infty$.*

Proof. By the given restriction, separate planning of each $S^R|_{a_j}$ independently maintains feasibility of S^R. We denote by $m_{j1}, m_{j2}, \ldots, m_{jx}$ the messages appearing in $S^R_{a_j}$. We define a complete undirected auxiliary graph $G' = (V', E')$ on the node set $V' = \{p_j\} \cup \{s_{j1}, s_{j2}, \ldots, s_{jx}\} \cup \{t_{j1}, \ldots, t_{jx}\}$ with edges (u, v) having length $d_G(u, v)$.

For $\underline{\kappa = 1}$, the schedule $\text{Opt}(S^R)|_{a_j}$ corresponds to a Hamiltonian path H in G' of minimum length, starting in p_j, subject to the condition that for each message m_{ji} the visit of its source s_{ji} is directly followed by a visit of its destination t_{ji}. We can lower bound the length of H with the total length of a spanning tree $T' = (V', E(T)') \subseteq G'$ as follows: Starting with an empty graph on V' we first add all edges (s_{ji}, t_{ji}). Following the idea of Kruskal [23], we add edges from $\{p_j\} \times \{s_{j1}, \ldots, s_{jx}\} \cup \{t_{j1}, \ldots, t_{jx}\} \times \{s_{j1}, \ldots, s_{jx}\}$ in increasing order of their lengths, disregarding any edges which would result in the creation of a cycle. Now a DFS-traversal of T' starting from p_j visits any edge (s_{ji}, t_{ji}) in both directions. Whenever we cross such an edge from s_{ji} to t_{ji}, we add $(a_j, s_{ji}, m_{ji}, +), (a_j, t_{ji}, m_{ji}, -)$ as a suffix to the current schedule $\text{Alg}(S^R)|_{a_j}$. We get an overall cost of $\text{cost}(\text{Alg}(S^R)|_{a_j}) \leq 2 \cdot \sum_{e \in E(T')} l_e \leq 2 \cdot \sum_{e \in H} l_e = 2 \cdot \text{cost}(\text{Opt}(S^R)|_{a_j})$.

For $\underline{\kappa = \infty}$, the idea is to first collect all messages by traversing a spanning tree (with cost $\leq 2 \cdot \text{cost}(\text{Opt}(S^R)|_{a_j})$) and then delivering all of them in a metric TSP path fashion (with cost $\leq \frac{3}{2} \cdot \text{cost}(\text{Opt}(S^R)|_{a_j})$). ◀

▶ Remark. If we assume as an additional property that the agent returns to its starting position p_j (as for example in the result of Theorem 7), we can get a better approximation for the case $\kappa = 1$. Instead of traversing a spanning tree twice, we can model this as the *stacker-crane problem* for which a polynomial-time 1.8-approximation is known [16].

4 Coordination

In this section, we focus on the *Coordination* aspect of WeightedDelivery. We assume that collaboration and planning are taken care of. More precisely, we are given a sequence containing the complete *fixed* schedule S^- of all actions $(\_, s_i, m_i, +), \ldots, (\_, h, m_i, -)$, $\ldots, (\_, t_j, m_j, -)$, but without an assignment of the agents to the actions. Coordination is the task of assigning agents to the given actions. Even though coordination appears to have the flavor of a matching problem, it turns out to be NP-hard to optimally match up agents with the given actions. This holds for any capacity, in particular for $\kappa = 1$. The latter, however, has a polynomial-time solution if all agents have uniform weight.

4.1 NP-Hardness for Planar Graphs

We give a reduction from planar 3SAT: From a given planar 3SAT formula F we construct an instance of WeightedDelivery that allows a schedule S with "good" energy $\text{cost}(S)$ if and only if F is satisfiable.

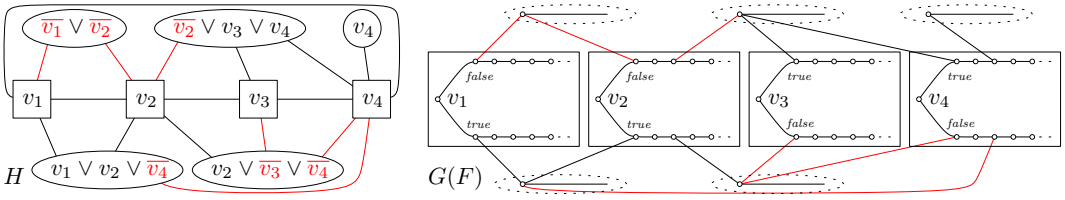


Figure 4 (left) A restricted plane embedding of a 3CNF F which is satisfied by $(v_1, v_2, v_3, v_4) = (true, false, false, true)$. (right) Its transformation to the corresponding delivery graph.

Planar 3SAT. Let F be a three-conjunctive normal form (3CNF) with x boolean variables $V(F) = \{v_1, \ldots, v_x\}$ and y clauses $C(F) = \{c_1, \ldots, c_y\}$. Each clause is given by a subset of at most three literals of the form $l(v_i) \in \{v_i, \overline{v_i}\}$. We define a corresponding graph $H(F) = (N, A)$ with a node set consisting of all clauses and all variables ($N = V(F) \cup C(F)$). We add an edge between a clause c and a variable v, if v or $\overline{v}$ is contained in c. Furthermore we add a cycle consisting of edges between all pairs of consecutive variables, i.e., $A = A_1 \cup A_2$, where $A_1 = \{\{c_i, v_j\} \mid v_j \in c_i \text{ or } \overline{v_j} \in c_i\}$, $A_2 = \left\{\left\{v_j, v_{(j \bmod x)+1}\right\} \mid 1 \leq j \leq x\right\}$. We call F *planar* if there is a plane embedding of $H(F)$. The *planar 3SAT* problem of deciding whether a given planar 3CNF F is satisfiable is known to be NP-complete. Furthermore the problem remains NP-complete *if at each variable node* the plane embedding is required to have all arcs representing positive literals on one side of the cycle A_2 and all arcs representing negative literals on the other side of A_2 [26]. We will use this *restricted version* in our reduction and assume without loss of generality that the graph $H(F) \setminus A_2$ is connected and that $H(F)$ is a simple graph (i.e. each variable appears at most once in every clause).

Building the Delivery Graph. We first describe a way to transform any planar 3CNF graph $H(F)$ into a planar delivery graph $G = G(F)$, see Figure 4.

We transform the graph in five steps: First we delete all edges of the cycle A_2, but we keep in mind that at each variable node all positive literal edges lie on one side and all negative literal edges on the other side. Secondly let $\deg_{H(F),A_1}(v)$ denote the remaining degree of a variable node v in H and surround each variable node by a *variable box*. A variable box contains two paths adjacent to v on which internally we place $\deg_{H(F),A_1}(v)$ copies of v: One path (called henceforth the *true-path*) contains all nodes having an adjacent positive literal edge, the other path (the *false-path*) contains all nodes having an adjacent negative literal edge. In a next step, we add a single node between any pair of node copies of the previous step. As a fourth step, we want all paths to contain the same number of nodes, hence we fill in nodes at the end of each path such that every path contains exactly $2y \geq 2\deg_{H(F),A_1}(v)$ internal nodes. Thus each variable box contains a variable node v, an adjacent *true*-path (with internal nodes $v_{true,1}, \ldots, v_{true,2y-1}$ and a final node $v_{true,2y}$) and a respective *false*-path. Finally for each clause node c we add a new node c' which we connect to c. The new graph $G(F)$ has polynomial size and all the steps can be implemented in such a way that $G(F)$ is planar.

Messages, Agents and Weights. We are going to place one *clause message* on each of the y clause nodes and a *literal message* on each of the $2x$ paths in the variable boxes for a total of $4xy$ messages. More precisely, on each original clause node c we place exactly one clause message which has to be delivered to the newly created node c'. Furthermore we place a literal message on every internal node $v_{true,i}$ of a *true*-path and set its target to $v_{true,i+1}$ (same for the *false*-path). We set the length of all edges connecting a source to its target to 1.

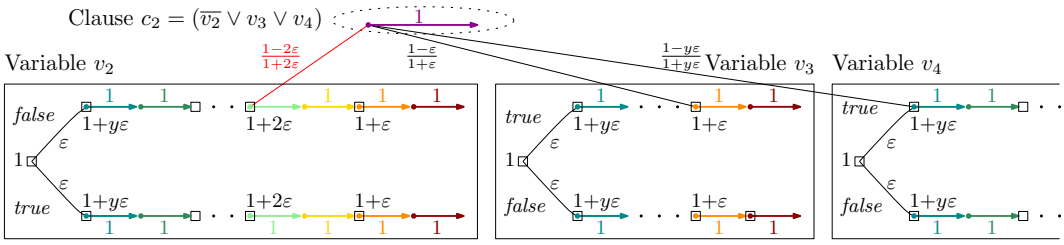


Figure 5 Agent positions (□) and weights (in black); Messages (→) and edge lengths (in color).

Next we describe the locations of the agents in each variable box. We place one *variable agent* of weight 1 on the variable node v. The length of the two adjacent edges are set to ε, where $\varepsilon := (8xy)^{-2}$. Furthermore we place y *literal agents* on each path: The i-th agent is placed on $v_{true,2(y-i)}$ (respectively $v_{false,2(y-i)}$) and gets weight $1 + i\varepsilon$. It remains to set the length of edges between clause nodes and internal nodes of a path. By construction the latter is the starting position of an agent of uniquely defined weight $1 + i\varepsilon$; we set the length of the edge to $\frac{1-i\varepsilon}{1+i\varepsilon}$. For an illustration see Figure 5, where each agent's starting location is depicted by a square and each message is depicted by a colored arrow.

Reduction. The key idea of the reduction is that for each variable u, the corresponding variable box contains a variable agent who can *either* deliver all messages on the *true*-path (thus setting the variable to true), *or* deliver all messages on the *false*-path (thus setting the variable to false). Assume u is set to true. If u is contained in a clause c, then on the adjacent node $v_{true,i}$ there is a (not yet used) literal agent. Intuitively, this agent was *freed by the variable agent* and can thus be sent to deliver the clause-message. If $\overline{u}$ is contained in c, the corresponding literal on the *false*-path can't be sent to deliver the clause message, since it needs to transport messages along the *false*-path.

There is such a feasible schedule Sol of the agents in $G(F)$ if and only if there is a satisfiable assignment (a solution) for the variables of a 3CNF F. Its total (energy) cost is $\text{cost}(\text{Sol}) := 4xy + 2y + x(y^2 + y + 1)\varepsilon$ ([6, Lemma 12]). Furthermore, we can show that *any* schedule S which doesn't correspond to a satisfiable variable assignment has cost $\text{cost}(S) > \text{cost}(\text{Sol})$ ([6, Lemma 13, 14 and 15]). This is true independent of whether S adheres to the schedule without agents S^- or not and holds for any capacity κ.

Fixed Sequence (Schedule without Agent Assignment). It remains to fix a sequence S^- that describes the schedule Sol described in the reduction idea but which does not allow us to infer a satisfiable assignment: This is the case for any S^- consisting of consecutive pairs $(\_, s_i, m_i, +), (\_, t_i, m_i, -)$ such that if m_i lies to the left of m_j on some *true*- or *false*-path, it precedes m_j in the schedule.

▶ **Theorem 11.** Coordination *of* WeightedDelivery *is* NP*-hard on planar graphs for all capacities* κ*, even if we are given prescribed collaboration and planning.*

4.2 Polynomial-time Algorithm for Uniform Weights and Unit Capacity

Note that Coordination is NP-hard even for capacity $\kappa = 1$. Next we show that this setting is approachable once we restrict ourselves to uniform weights.

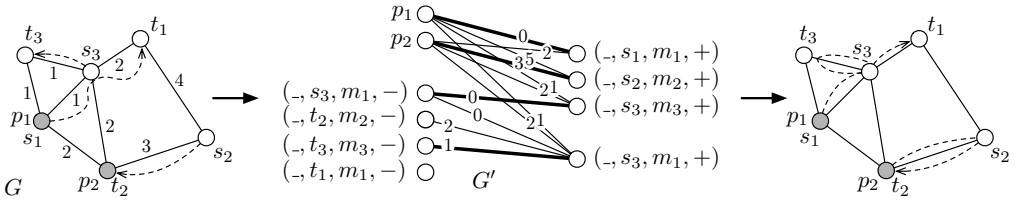


Figure 6 Illustration of the coordination of the schedule $S = (\_, s_1, m_1, +), (\_, s_2, m_2, +), (\_, s_3, m_1, -), (\_, s_3, m_3, +), (\_, t_2, m_2, -), (\_, t_3, m_3, -), (\_, s_3, m_1, +), (\_, t_1, m_1, -)$. (left) Instance with 3 messages and 2 agents of uniform weight. (center) Equivalent weighted bipartite matching problem G'. (right) The resulting trajectories of the agents.

▶ **Theorem 12.** *Given collaboration and planning in the form of a complete schedule with missing agent assignment, Coordination of* WeightedDelivery *with capacity* $\kappa = 1$ *and agents having uniform weights can be solved in polynomial time.*

Proof. As before, denote by $S^- = (\_, s_i, m_i, +), \ldots, (\_, h, m_i, -), \ldots, (\_, t_j, m_j, -)$ the prescribed schedule without agent assignments. Since all agents have the same uniform weight w, the cost $\text{cost}(S)$ of any feasible schedule S is determined by $\text{cost}(S) = w \cdot \sum_{j=1}^{k} d_j$. Hence at a pick-up action $(\_, q, m_i, +)$ it is not so much important *which* agent picks up the message as *where / how far* it comes from.

Because we have capacity $\kappa = 1$, we know that the agent has to come from either its starting position or from a preceding drop-off action $(\_, p, m_j, -) \in S^-$. This allows us to model the problem as a weighted bipartite matching, see Figure 6 (center). We build an auxiliary graph $G' = (A \cup B, E'_1 \cup E'_2)$. A maximum matching in this bipartite graph will tell us for every pick-up action in B, where the agent that performs the pick-up action comes from in A. Let $A := \{p_1, \ldots, p_k\} \cup \{(\_, *, *, -)\}$ and $B := \{(\_, *, *, +)\}$. We add edges between all agent starting positions and all pick-ups, $E'_1 := \{p_1, \ldots, p_k\} \times \{(\_, q, m, +) \mid (\_, q, m, +) \in B\}$ of weight $d_G(p_i, q)$. Furthermore, we add edges between drop-offs and all subsequent pick-ups $E'_2 := \{((\_, p, m_j, -), (\_, q, m_i, +)) \mid (\_, p, m_j, -) < (\_, q, m_i, +) \text{ in } S^-\}$ of weight $d_G(p, q)$.

A maximum matching of minimum cost in G' captures the optimal assignment of agents to messages and can be found by solving the classic *assignment problem*, a special case of the *minimum cost maximum flow problem*. Both of these problems can be solved in polynomial time for instance using the *Hungarian method* [24] or the *successive shortest path algorithm* [14], respectively. The cost of this optimum matching corresponds to the cost of the agents moving around without messages. The cost of the agents while carrying the messages can easily be added: Consider the schedule S^- restricted to a message m_i. This subsequence $S^-|_{m_i}$ is a sequence of pairs of pick-up/drop-off actions $((\_, q, m_i, +), (\_, p, m_i, -))$, and in every pair the message is brought from q to p on the shortest path, so we add $\sum d_G(q, p)$. Concatenating these piecewise shortest paths gives the trajectory of each agent in the optimum solution, as illustrated in Figure 6 (right). ◀

Our algorithm is remotely inspired by a simpler problem at the ACM ICPC world finals 2015 [19]. The official solution is pseudo-polynomial [20], Austrin and Wojtaszczyk [3] later sketched a min-cost bipartite matching solution.

5 Approximation Algorithm

We have seen in Section 2 that the cost of an optimal schedule Opt for WeightedDelivery can be approximated to within a factor of two by restricting ourselves to schedules S^R in

which every message is only transported by a single agent, with exactly one pick-up and one drop-off (Theorem 7). Let $\textsc{Opt}^R$ be an optimal restricted schedule. Given an auxiliary graph G' on a vertex set consisting of all agent positions and all message source and destination nodes, we construct in polynomial time a minimum tree cover of G' from which be build a schedule S^* in which agents travel at most twice the distance of their counterparts in $\textsc{Opt}^R$ (similarly to Theorem 10). Here we require capacity $\kappa = 1$. In order for our method to work, we need to be indifferent between different weights; we achieve this by boosting each agent's weight w_j to $\max w_i$, resulting in an additional loss in the approximation factor of $\max \frac{w_i}{w_j}$:

▶ **Theorem 13.** *There is a polynomial-time* $(4 \max \frac{w_i}{w_j})$*-approximation algorithm for* WeightedDelivery *with capacity* $\kappa = 1$.

References

1. Julian Anaya, Jérémie Chalopin, Jurek Czyzowicz, Arnaud Labourel, Andrzej Pelc, and Yann Vaxès. Convergecast and broadcast by power-aware mobile agents. *Algorithmica*, 74(1):117–155, 2016. doi:10.1007/s00453-014-9939-8.
2. David L. Applegate, Robert E. Bixby, Vasek Chvatal, and William J. Cook. *The Traveling Salesman Problem: A Computational Study (Princeton Series in Applied Mathematics)*. Princeton University Press, Princeton, NJ, USA, 2007.
3. Per Austrin and Jakub Onufry Wojtaszczyk. ACM ICPC World Finals 2015 solution sketches. http://www.csc.kth.se/~austrin/icpc/finals2015solutions.pdf, 2015.
4. 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.
5. A. Bärtschi, J. Chalopin, S. Das, Y. Disser, B. Geissmann, D. Graf, A. Labourel, and M. Mihalák. Collaborative Delivery with Energy-Constrained Mobile Robots, arXiv preprint, 2016. arXiv:1608.08500.
6. 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 preprint, 2016. URL: https://arxiv.org/abs/1610.02361.
7. Davide Bilò, Yann Disser, Luciano Gualà, Matús Mihalák, Guido Proietti, and Peter Widmayer. Polygon-constrained motion planning problems. In *9th International Symposium on Algorithms and Experiments for Sensor Systems, Wireless Networks and Distributed Robotics ALGOSENSORS'13*, pages 67–82, 2013. doi:10.1007/978-3-642-45346-5_6.
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. Nicos Christofides. Worst-case analysis of a new heuristic for the travelling salesman problem. Technical Report 388, Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh, PA, February 1976.
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 Ellis L. Johnson. Matching, euler tours and the chinese postman. *Mathematical Programming*, 5(1):88–124, 1973. doi:10.1007/BF01580113.

14 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.

15 Robert W. Floyd. Algorithm 97: shortest path. *Communications of the ACM*, 5(6):345, 1962.

16 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.

17 Greg N. Frederickson, Matthew S. Hecht, and Chul E. Kim. Approximation algorithms for some routing problems. In *Proceedings of the 17th Annual Symposium on Foundations of Computer Science (FOCS)*, 1976.

18 J. A. Hoogeveen. Analysis of christofides' heuristic: Some paths are more difficult than cycles. *Operations Research Letters*, 10(5):291–295, July 1991. doi:10.1016/0167-6377(91)90016-I.

19 ACM ICPC. World Finals 2015 Problems, Task C: Catering. https://icpc.baylor.edu/worldfinals/problems, May 2015.

20 ICPCNews. ACM ICPC 2015 problem catering. https://www.youtube.com/watch?v=WxCgcI18_d8, May 2015.

21 Alon Itai, Christos H Papadimitriou, and Jayme Luiz Szwarcfiter. Hamilton paths in grid graphs. *SIAM Journal on Computing*, 11(4):676–686, 1982.

22 Marek Karpinski, Michael Lampis, and Richard Schmied. New inapproximability bounds for tsp. *Journal of Computer and System Sciences*, 81(8):1665–1677, 2015.

23 Joseph B. Kruskal, Jr. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical Society*, 7:48–50, 1956.

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

25 Chung-Lun Li, S. Thomas McCormick, and David Simchi-Levi. The point-to-point delivery and connection problems: Complexity and algorithms. *Discrete Applied Mathematics*, 36(3):267–292, 1992.

26 David Lichtenstein. Planar formulae and their uses. *SIAM Journal on Computing*, 11(2):329–343, 1982. doi:10.1137/0211025.

27 Stephen Warshall. A theorem on boolean matrices. *Journal of the ACM (JACM)*, 9(1):11–12, 1962.

28 Pawel Winter. Steiner problem in networks: A survey. *Networks*, 17(2):129–167, 1987. doi:10.1002/net.3230170203.