


Two Results on LPT: A Near-Linear Time Algorithm and Parcel Delivery Using Drones

L. Sunil Chandran   

Indian Institute of Science, Bengaluru, India

Rishikesh Gajjala   

Indian Institute of Science, Bengaluru, India

Shravan Mehra 

Indian Institute of Science, Bengaluru, India

University of Birmingham, UK

Saladi Rahul   

Indian Institute of Science, Bengaluru, India

Abstract

The focus of this paper is to increase our understanding of the *Longest Processing Time First (LPT)* heuristic. LPT is a classical heuristic for the fundamental problem of uniform machine scheduling. For different machine speeds, LPT was first considered by Gonzalez et al. (*SIAM J. Comput.* 6(1):155–166, 1977). Since then, extensive work has been done to improve the approximation factor of the LPT heuristic. However, all known implementations of the LPT heuristic take $O(mn)$ time, where m is the number of machines and n is the number of jobs. In this work, we come up with the first *near-linear time* implementation for LPT. Specifically, the running time is $O((n+m)(\log^2 m + \log n))$. Somewhat surprisingly, the result is obtained by mapping the problem to dynamic maintenance of lower envelope of lines, which has been well studied in the computational geometry community.

Our second contribution is to analyze the performance of LPT for the *Drones Warehouse Problem (DWP)*, which is a natural generalization of the uniform machine scheduling problem motivated by drone-based parcel delivery from a warehouse. In this problem, a warehouse has multiple drones and wants to deliver parcels to several customers. Each drone picks a parcel from the warehouse, delivers it, and returns to the warehouse (where it can also get charged). The speeds and battery lives of the drones could be different, and due to the limited battery life, each drone has a bounded range in which it can deliver parcels. The goal is to assign parcels to the drones so that the time taken to deliver all the parcels is minimized. We prove that the natural approach of solving this problem via the LPT heuristic has an approximation factor of ϕ , where $\phi \approx 1.62$ is the golden ratio.

2012 ACM Subject Classification Theory of computation

Keywords and phrases Scheduling, Approximation algorithms, Fine-grained complexity

Digital Object Identifier 10.4230/LIPIcs.FSTTCS.2024.17

Related Version *Full Version*: <https://arxiv.org/abs/2407.16323>

Funding *Shravan Mehra*: This work was supported by SERB Core Research Grant (CRG/2022/006770): “Bridging Quantum Physics with Theoretical Computer Science and Graph Theory”.

Saladi Rahul: This work was supported by the Walmart Center for Tech Excellence at IISc (CSR Grant WMGT-230001).

1 LPT heuristic for uniform scheduling

Uniform machine scheduling with the minimum makespan objective is a fundamental problem. In this problem, we are given a set of n jobs (not necessarily of the same size) and a set of m machines (not necessarily of the same speeds). The goal is to schedule the n jobs on m



© L. Sunil Chandran, Rishikesh Gajjala, Shravan Mehra, and Saladi Rahul; licensed under Creative Commons License CC-BY 4.0

44th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2024).

Editors: Siddharth Barman and Sławomir Lasota; Article No. 17; pp. 17:1–17:15



Leibniz International Proceedings in Informatics

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

machines so that the time required to execute the schedule (makespan) is minimised. This is an NP-hard problem even for two machines [13] of the same speed, but polynomial time approximation schemes (PTASs) are known [20, 21].

A commonly studied heuristic for this problem is the *Longest Processing Time First* (LPT) heuristic. In the LPT heuristic, each job is assigned one by one, in non-increasing order of size, so that every job is assigned to a machine where it will be completed earliest (with ties being broken arbitrarily). Note that a machine might already have some jobs assigned to it and the execution of the current job happens only after finishing the already assigned jobs.

More intricate algorithms were designed in the literature to get a good approximation factor for uniform scheduling. For example, Horowitz and Sahni gave an exact dynamic programming algorithm which runs in exponential time [22]. When there are only two machines, they could build upon this algorithm to obtain a Polynomial-time approximation scheme (PTAS). Later, Hochbaum and Shmoys gave a PTAS when all the machines had identical speeds [20]. This was later extended to obtain a PTAS for the uniform scheduling problem (USP) [21].

However, the LPT algorithm remains popular in practice due to its simplicity and scalability (compared to the PTAS-type results which are relatively complicated and have expensive running time). As a result, there has been a long line of research on improving the approximation ratio of the LPT algorithm for USP. In two independent works, the ratio of the LPT algorithm was improved by Dobson [10] to $\frac{19}{12}$ and by Friesen [12] to $\frac{5}{3}$. Kovacs [28] further improved the approximation factor of the LPT algorithm to 1.58 and proved that the LPT algorithm cannot give an approximation factor better than 1.54.

2 First result: A near-linear time implementation of LPT

In spite of all the focus in the literature on adapting LPT to various settings of machine scheduling and analyzing its approximation factor, to the best of our knowledge, there has been no work on fast implementation of LPT. The known implementation of the LPT heuristic takes $O(mn)$ time (via the naive approach). In this work, we give the first near-linear time implementation of the LPT heuristic.

► **Theorem 1.** *There is an $O((n + m)(\log^2 m + \log n))$ time implementation of the LPT heuristic.*

For a set of given lines in 2-D, the *lower envelope* is the point-wise minimum of the lines (a more formal definition will follow later). In the dynamic maintenance of the lower envelope problem, in each step, a new line is added (or removed), as shown in Figure 2, and one has to maintain the lower envelope with a small update time. This has been well-studied in the computational geometry community. We establish a connection from LPT to the dynamic maintenance of the lower envelope of lines to prove Theorem 1 in Section 5.

3 Second result: LPT for the drones warehouse problem (DWP)

Our second contribution is to analyze the performance of LPT for the *Drones warehouse problem (DWP)* (formally defined in Section 3.1) which is a natural generalization of the uniform scheduling problem to drone-based parcel delivery from a warehouse.

Vehicle routing [14, 47] is a classical problem in which parcel deliveries are done by a single truck or a collection of trucks. Researchers have explored variations with different vehicle velocities [15] or scenarios where each parcel can only be delivered by a specific subset of

vehicles [49]. With the advent of drones, a generalization of the vehicle routing problem has been studied in the literature in which a truck is carrying drones along with it. The drones pick up parcels from the truck, deliver the package and return to the truck (the truck might have now moved to a different location). This problem is more challenging than the traditional vehicle routing problem [8]. Several MIP (mixed integer programming) formulations and heuristics have been used to solve this problem [38, 2]. Theoretical guarantees have also been proved for this problem by Carlsson and Song using geometric methods [8].

The transition towards using only drones like in DWP (instead of trucks with drones) is evident in the gradual shift within the research community, as it is a more sustainable option for the future. Extensive efforts have been dedicated to developing algorithms for scheduling drones under various constraints [44]. Some MIP formulations were studied to minimize various objectives like the number of drones [18, 24] and bio-inspired algorithms were used to manage large fleets of drones [40]. As drones have a limited battery life, considerations for fuel stations were explored in [25]. Further, the drones might not all have the same features like speed and battery life and this was taken into account in [46]. For a comprehensive review of research in several related problems involving only drones, the reader can refer to the survey presented in [41]. There has also been a lot of work by the multi-agent community for scheduling [26], pathfinding [9, 19] and coordinating drones [39, 36, 6, 48].

In this paper, we consider the problem where a warehouse wants to use *drones* to deliver a large number of parcels to customers around it. Due to limited battery life, each drone has a restricted range around the warehouse in which it can deliver parcels. Also, depending on the manufacturer, the speed of each drone can vary. We will now formally define the DWP problem, state the result obtained by applying the LPT heuristic for DWP and provide a high-level overview of the analysis. We also provide a detailed literature review on the use of LPT for several other machine scheduling problems and connect it to our result on DWP in Section 4.

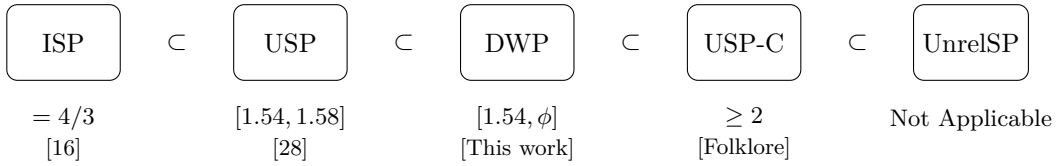
3.1 The drones warehouse problem (DWP)

For the sake of better readability, we deviate from the notation used in scheduling literature. The warehouse has a set of m drones $\mathcal{D} = \{D_1, D_2 \dots D_m\}$ and a set of n parcels $\mathcal{P} = \{P_1, P_2 \dots P_n\}$. The parcels in set \mathcal{P} need to be delivered by drones in set \mathcal{D} . Each drone can pick up one parcel at a time from the warehouse, deliver it and return to the warehouse. For all $1 \leq j \leq n$, let the distance at which parcel P_j needs to be delivered be $\ell_j/2$. So each drone must travel a distance of ℓ_j in total to deliver the parcel P_j and come back to the warehouse.

Additionally, the drones have a limited battery life which can be different for each drone. Let d_i be the distance which drone D_i can travel, for all $1 \leq i \leq m$. Therefore, for a parcel P_j to be delivered by a drone D_i , it must be the case that $\ell_j \leq d_i$. The speed at which the drone D_i travels is v_i , for all $1 \leq i \leq m$. After each delivery, the drone recharges its battery in the warehouse, and for the sake of simplicity we assume the time taken to recharge is negligible. Our goal is to assign each parcel to a drone such that the time taken to execute the schedule is minimised.

More precisely, we define a *valid schedule* $f : \mathcal{D} \rightarrow 2^{\mathcal{P}}$ (power set of \mathcal{P}) such that the following properties hold

- Each parcel is assigned to exactly one drone, i.e., $f(D_i) \cap f(D_j) = \emptyset$ for all $i \neq j$ and $\bigcup_{D \in \mathcal{D}} f(D) = \mathcal{P}$.
- Each drone must be able to deliver the parcel assigned to it, i.e., $\ell_i \leq d_j$ for all $j \in [m]$ and $i : P_i \in f(D_j)$.



■ **Figure 1** Landscape of the approximation ratio of the LPT heuristic for machine scheduling problems and our drone warehouse problem (DWP). The relation $A \subset B$ in the figure implies that A is a special case of B . Therefore, the approximation factor increases from left to right in the figure. The interval $[a, b]$ means that the approximation ratio of the LPT heuristic is at least a and at most b . As there is no total order among jobs in UnrelSP, LPT is not applicable for UnrelSP.

Our goal is to find a *valid schedule* such that $T(f)$ is minimised where

$$T(f) = \max_{j \in [m]} \sum_{i: P_i \in f(D_j)} \frac{\ell_i}{v_j}$$

We assume that there is at least one drone capable of delivering all parcels. Otherwise, there would be no valid solution (this can be checked in linear time).

3.2 Our results and techniques

We implement the LPT algorithm with additional battery life constraints to solve DWP in near-linear time. Our key contribution is to prove that this algorithm always returns a solution which has delivery time at most ϕ times the optimal solution, where $\phi \approx 1.62$ is the golden ratio. We summarize our results and compare them with previous work in Figure 1 (discussed in more detail in Section 4)

► **Theorem 2.** *There is a ϕ -approximation algorithm for the DWP problem where $\phi = \frac{1+\sqrt{5}}{2}$ is the golden ratio. The algorithm runs in $O((n+m)(\log^2 m + \log n))$ time, where m and n are the number of drones and parcels, respectively.*

At a high level, our proof is inspired by the analysis of the LPT algorithm for the uniform scheduling problem (USP) [28]. However, the constraint of battery life makes our problem significantly more challenging. As the total distance travelled by the drones is the same for any valid schedule, if some drone in the LPT algorithm travels more distance than its counterpart in the optimal assignment, then some other drone in the LPT algorithm must travel lesser distance than its counterpart in the optimal assignment as a compensation. However, if we assume that the approximation factor is greater than ϕ , we can create instances for which such compensation does not occur, which leads to a contradiction. The proof requires ideas such as (a) working with a minimal instance, (b) removing the parcel which is closest to the warehouse from the schedule, (c) classifying the parcels into three categories based on their distance and (d) truncating the parcel distances. We give the complete proof in Section 6.

4 Related work on machine scheduling

We will now give a detailed literature review of the use of LPT for machine scheduling and connect our problem DWP with the other variants.

4.1 Uniform machines scheduling problem (USP)

In the *uniform machines scheduling problem*, let $\mathcal{P} = \{P_1, P_2, \dots, P_n\}$ be a set of n jobs of size $\{\ell_1, \ell_2, \dots, \ell_n\}$ respectively and $\mathcal{D} = \{D_1, D_2, \dots, D_m\}$ be a set of m machines of speed $\{v_1, v_2, \dots, v_m\}$ respectively. Our goal is to schedule the n jobs to the m machines so that the completion time of the schedule is minimised. We will now formally define the problem.

We define a *schedule* as a function $f : \mathcal{D} \rightarrow 2^{\mathcal{P}}$ (power set of \mathcal{P}) where $f(D_i)$ represents the set of all jobs assigned to machine D_i . We call a *schedule* a *valid schedule* if each job is assigned to exactly one machine, i.e., f is a *valid schedule* if and only if $f(D_i) \cap f(D_j) = \emptyset$ for all $i \neq j$ and $\bigcup_{D \in \mathcal{D}} f(D) = \mathcal{P}$. Each schedule f has an associated completion time

$$T(f) = \max_{j \in [m]} \sum_{i: P_i \in f(D_j)} \frac{\ell_i}{v_j}$$

Our goal is to find a *valid schedule* f such that $T(f)$ is minimised.

4.2 ISP \subset USP \subset DWP

The special case of USP when all the machines have equal speed is called the *identical-machines scheduling problem (ISP)*. Therefore, we denote $\text{ISP} \subset \text{USP}$, where the notation $A \subset B$ means that A is a special case of B (See Figure 1) and therefore, a lower bound of A is also a lower bound for B and an upper bound of B is also an upper bound for A . Graham [16, 17] proved that the approximation factor of the LPT algorithm is $\frac{4}{3}$ for ISP. For USP, after a series of works, Kovacs [28] proved that the approximation factor of the LPT algorithm is at most 1.58 and at least 1.54. It is easy to see that USP is a special case of DWP ($\text{USP} \subset \text{DWP}$) when all battery lives are large enough to deliver all parcels. So, the approximation factor of LPT for DWP can not be better than 1.54. On the other hand, due to Theorem 2, we get that the LPT heuristic is a ϕ -approximation, which is one of the main contributions of this work.

4.3 DWP \subset USP-C \subset UnrelSP

A lot of work in the literature has been devoted to a generalization of USP, namely uniform scheduling problems with *processing constraints (USP-C)* [30]. We are given a set of jobs $\mathcal{J} = \{J_1, J_2, \dots, J_n\}$ and a set of machines \mathcal{M} , where each job J_j has a set of machines $\mathcal{M}_j \subseteq \mathcal{M}$ to which they can be assigned to. Different structural restrictions of \mathcal{M}_j lead to different models in USP-C, like the inclusive processing set [42, 35], nested processing set [37, 11], interval processing set [45, 23], and tree-hierarchical processing set restrictions [34, 33]. The goal is to assign each job to a machine to minimize the completion time.

DWP is also a special case of USP-C as each parcel can only be delivered by a subset of drones determined by its battery life. Therefore, $\text{DWP} \subset \text{USP-C}$. We emphasise that these two classes are strictly different (and DWP is also different from all known variants of USP-C, including nested intervals). Consider two machines and two jobs. Let the size of J_1, J_2 be $10, 10 + \epsilon$ respectively and the speeds of M_1, M_2 be $10, 10 + \epsilon$ respectively for $\epsilon > 0$. Moreover, assume that the job J_1 can only be done by M_2 , but J_2 can be done by both M_1 and M_2 . The LPT heuristic would take $\frac{20+\epsilon}{10+\epsilon}$ time, while the optimum assignment would take only $\frac{10+\epsilon}{10}$ time. This gives us an approximation ratio of 2 as ϵ approaches zero in this example (whether this ratio is optimal or not for the LPT heuristic on USP-C is an open question). We also note that this is not a valid lower bound instance for DWP as any drone which can do a job at a distance of $10 + \epsilon$ can also do the job at a distance of 10.

The unrelated Scheduling Problem (UnrelSP) is the scheduling problem in which the time taken by machine $D \in \mathcal{D}$ to complete job $P \in \mathcal{P}$ is determined by an arbitrary function $f : \mathcal{D} \times \mathcal{P} \rightarrow \mathbb{R}$. Note that USP-C \subset UnrelSP and furthermore the LPT heuristic is not applicable for UnrelSP as there is no total order among the jobs to sort. This finishes the description of Figure 1.

4.4 Other algorithms and optimization measures

A PTAS for ISP was given by Hochbaum and Shmoys [20]. This was later extended to obtain a PTAS for USP [21]. Due to the seminal result of Lenstra, Shmoys and Tardos [29], there is an LP-based 2-approximation algorithm for UnrelSP. This is also the best known approximation algorithm for USP-C. For a special case of USP-C (including DWP) with nested intervals, there is 4/3 approximation algorithm [32]. For more results on USP-C, we refer the reader to the latest survey [31]. We emphasise that despite knowing PTASs and other algorithms with a better approximation ratio, the LPT heuristic remains popular in practice due to its simplicity and scalability. This motivated researchers from the algorithms and operations research community to extensively study it as described in Figure 1.

Instead of optimizing the time taken to complete all jobs (makespan), other objectives like the average completion time [7], weighted-average completion time [22] and monotonicity and truthfulness have also been studied [5, 27, 4, 3].

5 Near linear time implementation

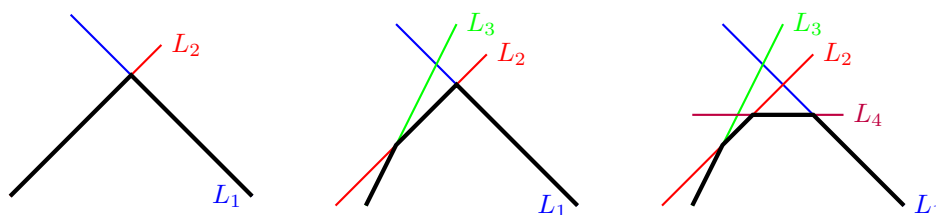
5.1 Longest processing time first (LPT)

A classical approach for the Uniform Scheduling problem is using a greedy algorithm called LPT scheduling which gives a 1.58-approximate solution [28]. First, we sort the jobs \mathcal{P} in decreasing order of their size and let P_1, P_2, \dots, P_n be the sorted sequence. Then we initialise T_j , the time taken by the j th machine to be zero for all $j \in [m]$. Now we assign the jobs sequentially from P_1 to P_n . We assign job P_i to a machine D_j for which the value of $T_j + (\ell_i/v_j)$ is minimum and we update the value T_j to $T_j + (\ell_i/v_j)$ for this specific j (see Algorithm 1).

■ **Algorithm 1** LPT algorithm in $O(nm)$ time.

Input: List of jobs \mathcal{P} and machines \mathcal{D} .
Output: Time required for LPT scheduling.
Sort \mathcal{P} in non-increasing order of size.
Initialise $T_j = 0$ for all $j \in [m]$
for i in $\{1 \dots n\}$ **do**
 $\alpha = \arg \min_{j \in [m]} \left(T_j + \frac{\ell_i}{v_j} \right)$
 $T_\alpha \leftarrow T_\alpha + \frac{\ell_i}{v_\alpha}$
end for
return $\max_{j \in [m]} T_j$

As $\arg \min_{j \in [m]} T_j + \frac{\ell_i}{v_j}$ can be found in $O(m)$ time, we can implement the above algorithm to run in $O(nm)$ time. To improve the run time of this algorithm, we implement a faster way to find $\arg \min_{j \in [m]} T_j + \frac{\ell_i}{v_j}$.



■ **Figure 2** Dynamic lower envelope of lines $L_1 : y = -x + 4$, $L_2 : y = x$, $L_3 : y = 2x$, $L_4 : y = 1$.

5.2 Dynamic Lower Envelope

One can visualise the step in which $\arg \min_{j \in [m]} T_j + \frac{\ell_i}{v_j}$ is computed in the following way:

Consider the m linear functions h_1, h_2, \dots, h_m where $h_j(x) = \frac{1}{v_j} \cdot x + T_j$. We need to find

the index of $j \in [m]$ for which $h_j(x) = \frac{1}{v_j} \cdot x + T_j$ is minimum at $x = \ell_i$. This is exactly the same as finding the line in the lower envelope for the h_1, h_2, \dots, h_m at $x = \ell_i$. Our idea now is to maintain a dynamic lower envelope data structure capable of inserting and deleting functions. We will now describe this formally.

Let \mathcal{H} be a set of functions where $h \in \mathcal{H}$ is of the form $h : \mathbb{R} \rightarrow \mathbb{R}$. Then the lower envelope is the function $h_{min} : \mathbb{R} \rightarrow \mathbb{R}$ such that $h_{min}(x) = \min_{h \in \mathcal{H}} h(x)$. We are interested in the case when these functions correspond to straight lines, i.e., they are of the form $h_i(x) = m_i x + c_i$. More particularly, we are interested in the problem of dynamically maintaining the lower envelope problem of lines, i.e., in each step, a new line is added (or removed), as shown in Figure 2, and one has to maintain the lower envelope with a small update time. This has been well-studied in the computational geometry community. From [43], there is a data structure to maintain a dynamic lower envelope of lines (further extended to dynamic lower envelope of pseudo lines in [1]):

- $\mathcal{H}.Insert(h)$: Adds the linear function h to set \mathcal{H} in $O(\log^2 |\mathcal{H}|)$ time.
- $\mathcal{H}.Delete(h)$: Removes the function h from set \mathcal{H} in $O(\log^2 |\mathcal{H}|)$ time.
- $\mathcal{H}.LowerEnvelope(x)$: Returns $h^* = \arg \min_{h \in \mathcal{H}} h(x)$ in $O(\log |\mathcal{H}|)$ time.

Using this, we can implement the LPT algorithm in $O((n+m) \log^2 m)$ time (Algorithm 2). We initialise the lower envelope data structure \mathcal{H} and store lines $h_j(x) = \frac{1}{v_j} x + 0$ (initially $T_j = 0$) for all $j \in [m]$. To assign parcel P_i to drone D_j , we remove the line $h_j(x) = \frac{1}{v_j} x + T_j$ from \mathcal{H} and replace it with $h'_j(x) = \frac{1}{v_j} x + T_j + \frac{\ell_i}{v_j}$. To find out which drone P_i is assigned to, we have to find $\arg \min_{j \in [m]} T_j + \frac{\ell_i}{v_j}$, which is the same as querying $\mathcal{H}.LowerEnvelope(\ell_i)$.

Observe that sorting \mathcal{P} takes $O(n \log n)$ time. We have called $\mathcal{H}.Insert(\cdot)$ $O(n+m)$ times and $\mathcal{H}.Delete(\cdot)$ $O(n)$ times. Therefore, the runtime of the algorithm is $O((n+m) \log^2 m + n \log n)$ or $O((n+m)(\log^2 m + \log n))$.

6 ϕ -approximation for the Drone Warehouse Problem

In this section, we will prove Theorem 2.

6.1 Algorithm

We use an algorithm similar to LPT but modify it slightly so that the battery constraints are respected. First, we sort the parcels \mathcal{P} in non-increasing order of their distance. Then, we initiate T_j , the time taken by the j th drone to be zero for all $j \in [m]$. We now assign

■ **Algorithm 2** LPT algorithm in $O((n+m)(\log^2 m + \log n))$ time.

Input: List of jobs \mathcal{P} and machines \mathcal{D} .
Output: Time required for LPT scheduling.
Sort \mathcal{P} in non-increasing order of size.
Initialise lower envelope data structure \mathcal{H}
for j in $\{1 \dots m\}$ **do**
 $h_j(x) = \frac{1}{v_j}x$
 $\mathcal{H}.Insert(h_j)$
end for
for i in $\{1 \dots n\}$ **do**
 $h_j = \mathcal{H}.LowerEnvelope(\ell_i)$
 Let $h_j(x) = \frac{1}{v_j}x + T_j$
 Set $h'_j(x) = \frac{1}{v_j}x + T_j + \frac{\ell_i}{v_j}$
 $\mathcal{H}.Delete(h_j)$
 $\mathcal{H}.Insert(h'_j)$
end for
return $\max_{j \in [m]} h_j(0)$

the parcels sequentially from P_1 to P_n . We assign parcel P_i to a drone D_j for which $\ell_i \leq d_j$ and the value of $T_j + (\ell_i/v_j)$ is minimum. We update the value T_j to $T_j + (\ell_i/v_j)$ (see Algorithm 3). It is easy to see that the running time of the LPT algorithm is $O(mn + n \log n)$

6.2 Implementation

Let us sort all the drones in decreasing order of their battery life. Observe that if drone D_j is capable of delivering parcel P_i , then all drones $D_{j'}, j' \leq j$ are capable of delivering parcel P_i . Therefore, we can represent all the drones capable of delivering P_i by a pointer ptr so that all drones $D_j, j \in \{1, \dots, ptr\}$ can deliver parcel P_i . Also, as the parcels are sorted in decreasing order of distance, the value of ptr will only increase in each iteration (as any drone capable of delivering P_i can also deliver $P_{i'}, i' > i$). We again use the lower envelope data structure to implement LPT. (see Algorithm 3).

Observe, that sorting \mathcal{P} and \mathcal{D} takes $O(n \log n + m \log m)$ time. We call $\mathcal{H}.Insert(\cdot)$ $O(n+m)$ times and $\mathcal{H}.Delete(\cdot)$ $O(n)$ times. As these function calls only use $O(\log^2 m)$ time, the above algorithm runs in $O((n+m)(\log^2 m + \log n))$ time.

6.3 Proof for ϕ -approximation

Simplifying steps

Assume that the parcels are sorted in decreasing order of distance, i.e., if $i < j$ then $\ell_i \geq \ell_j$ for all $i, j \in [n]$. Let Alg_I represent the *valid schedule* obtained from the LPT-algorithm and let Opt_I represent some fixed optimal *valid schedule* on instance $I = \{\mathcal{D}, \mathcal{P}\}$. We show that $\frac{T(Alg_I)}{T(Opt_I)} \leq \phi$. We will start by performing some simplifying steps on I .

► **Lemma 3.** *For any instance $I = \{\mathcal{D}, \mathcal{P}\}$, we can assume that $\ell_n = 1$, $T(Opt_I) = 1$ and no drone has battery life less than ℓ_n .*

■ **Algorithm 3** LPT algorithm for DWP.

Input: List of drones \mathcal{D} and parcels \mathcal{P} .
Output: Minimum time required to deliver all parcels.
Sort \mathcal{P} in non-increasing order of distance.
Sort \mathcal{D} in non-increasing order of battery life
Initialise lower envelope data structure \mathcal{H}
 $ptr \leftarrow 0$
for i in $\{1 \dots n\}$ **do**
 while $ptr < n$ and $d_{ptr+1} \geq \ell_i$ **do**
 $ptr \leftarrow ptr + 1$
 $h_{ptr}(x) = \frac{1}{v_{ptr}}x$
 $\mathcal{H}.Insert(h_{ptr})$
 end while
 $h_j = \mathcal{H}.LowerEnvelope(\ell_i)$
 Let $h_j(x) = \frac{1}{v_j}x + T_j$
 Set $h'_j(x) = \frac{1}{v_j}x + T_j + \frac{\ell_i}{v_j}$
 $\mathcal{H}.Remove(h_j)$
 $\mathcal{H}.Insert(h'_j)$
end for
return $\max_{j \in [m]} h_j(0)$

Proof. Observe that scaling all values $\{\ell_i\}_{i \in [n]}, \{d_j\}_{j \in [m]}$ by some constant α scales $T(Alg_I)$ and $T(Opt_I)$ by α . Similarly, scaling all values $\{v_j\}_{j \in [m]}$ by some constant β scales $T(Alg_I)$ and $T(Opt_I)$ by β^{-1} . However, this procedure does not affect the value of the approximation factor $\frac{T(Alg_I)}{T(Opt_I)}$. Therefore, we can choose values α, β such that $\ell_n = 1$ and $T(Opt_I) = 1$ (choosing $\alpha = \ell_n^{-1}, \beta = \ell_n^{-1}T(Opt_I)$ gives us the desired result). Also, as P_n is the smallest job, we can remove all drones which do not have enough battery life to deliver it as such drones would be empty in any schedule. ◀

6.3.1 Idea-1: Working with minimal instances

Our goal is to prove that $T(Alg_I) \leq \phi$ for all instances I . Towards a contradiction, assume that there exists an instance I for which $\frac{T(Alg_I)}{T(Opt_I)} > \phi$. Among all such contradicting instances, let \mathcal{I} be a contradicting instance which has the *minimum* number of parcels. We will sometimes drop the subscripts in $Alg_{\mathcal{I}}$ and $Opt_{\mathcal{I}}$ and write them as Alg and Opt , respectively, for simplicity.

6.3.2 Idea-2: A schedule without the last parcel

As $\mathcal{I} = \{\mathcal{D}, \mathcal{P}\}$ is a contradicting instance with the least number of parcels, it means that for any other instance \mathcal{I}' with fewer parcels that \mathcal{I} is not a contradicting instance. In particular, this is true for $\mathcal{I}' = \{\mathcal{D}, \mathcal{P} \setminus \{P_n\}\}$. Intuitively, this implies $T(Alg_{\mathcal{I}'}) \leq \phi$ and $T(Alg) > \phi$, and adding parcel P_n causes the increase in time. We will now prove this rigorously.

► **Definition 4.** Let Alg_0 be the schedule obtained by removing parcel P_n from the schedule Alg , i.e., Alg_0 is a schedule such that $Alg_0(D_i) = Alg(D_i) \setminus \{P_n\}$ for all $i \in [m]$.

17:10 LPT in Near-Linear Time and Parcel Delivery Using Drones

► **Definition 5.** Let $L_j(f)$ represent the total distance travelled by drone D_j in schedule f . Then,

$$L_j(f) = \sum_{i:P_i \in f(D_j)} \ell_i.$$

► **Definition 6.** The completion time of a drone in schedule f is the time taken by the drone to deliver all parcels assigned to it by the schedule f .

► **Lemma 7.** $\ell_n + L_i(\text{Alg}_0) = 1 + L_i(\text{Alg}_0) > \phi v_i$, for all $i \in [m]$.

Proof. We claim that $T(\text{Alg}_0) \leq \phi$. Suppose not. Then consider the instance $\mathcal{I}' = \{\mathcal{D}, \mathcal{P} \setminus \{P_n\}\}$ obtained from the minimal instance $\mathcal{I} = \{\mathcal{D}, \mathcal{P}\}$. Observe that the schedule $\text{Alg}_{\mathcal{I}'}$ and schedule Alg_0 are the same (as the assignment of the first $n-1$ parcels is independent of the n^{th} parcel). Therefore, $T(\text{Alg}_{\mathcal{I}'}) > \phi$. Also, observe that $T(\text{Opt}_{\mathcal{I}'}) \leq T(\text{Opt}_{\mathcal{I}}) = 1$. This implies that $\frac{T(\text{Alg}_{\mathcal{I}'})}{T(\text{Opt}_{\mathcal{I}'})} > \phi$ which contradicts the fact that \mathcal{I} is an instance with the least number of parcels such that $\frac{T(\text{Alg}_{\mathcal{I}})}{T(\text{Opt}_{\mathcal{I}})} > \phi$.

Now as $T(\text{Alg}) > \phi$ and $T(\text{Alg}_0) \leq \phi$, this means that only the drone which delivers P_n has completion time greater than ϕ . Also as the LPT algorithm assigns parcels to drones which have the least completion time, this implies that assigning P_n to any drone D_i in Alg_0 would result in its completion time being greater than ϕ . Note that P_n can be assigned to any drone as all drones have battery life at least ℓ_n (from Lemma 3).

Therefore, $(L_i(\text{Alg}_0) + \ell_n)/v_i > \phi$. As $\ell_n = 1$, it follows that $L_i(\text{Alg}_0) + 1 > \phi v_i$. ◀

6.3.3 Idea-3: Classification of parcels

We classify parcels for which the drone travels a distance in the range $[1, \phi)$ as *small* jobs, $[\phi, 2)$ as *medium* jobs and $[2, \infty)$ as *large* jobs. We first define a rounding function $R(x)$ such that

$$R(x) = \begin{cases} 1 & \text{if } x \in [1, \phi) \\ 1.5 & \text{if } x \in [\phi, 2) \\ \lfloor x \rfloor & \text{if } x \in [2, \infty) \end{cases}$$

Note that R is a function such that

$$x \geq R(x) \geq x/\phi \text{ for all } x \geq 1$$

Define $L'_i(f) = \sum_{j:P_j \in f(D_i)} R(\ell_j)$. The motivation for defining $R(x)$ is that the value of $L'_i(f)$ can only be integer multiples of 0.5 whereas $L_i(f)$ can take any arbitrary value depending on the input.

6.3.4 Idea-4: Discretizing Distances

Ideally, we would like to show that each drone in Alg_0 travels more distance than its counterpart in Opt . This would show that the total distance travelled by drones in Alg_0 is greater than that of drones in Opt which is a contradiction as fewer parcels have been delivered in Alg_0 than in Opt . Unfortunately, it turns out that $L_i(\text{Alg}_0)$ can be lesser than $L_i(\text{Opt})$. Instead, we show that $L'_i(\text{Opt}) \leq L'_i(\text{Alg}_0)$ for all $i \in [m]$ and arrive at a similar contradiction by analogous argument.

► **Lemma 8.** $L'_i(\text{Alg}_0) + \phi - 1 > v_i$ for all $i \in [m]$

Proof. From Lemma 7, we get that

$$v_i < \frac{1 + L_i(\text{Alg}_0)}{\phi} \leq \frac{1}{\phi} + L'_i(\text{Alg}_0) = \phi - 1 + L'_i(\text{Alg}_0)$$

The second inequality and the third equality follow from $x/\phi \leq R(x)$ and ϕ being the golden ratio, respectively. ◀

► **Lemma 9.** *If a drone D_i has a medium job assigned to it, then $L'_i(\text{Alg}_0) + 0.5 > v_i$*

Proof. Let a medium-sized job of size y be assigned to drone D_i . Then,

$$L'_i(\text{Alg}_0) = R(y) + \sum_{z \in \text{Alg}_0(D_i) \setminus \{y\}} R(z)$$

Since $y \in [\phi, 2)$, we get $R(y) = 1.5 > y - 0.5$, and hence,

$$L'_i(\text{Alg}_0) > y - 0.5 + \sum_{z \in \text{Alg}_0(D_i) \setminus \{y\}} R(z) \geq y - 0.5 + \frac{L_i(\text{Alg}) - y}{\phi}$$

Using Lemma 7 and $y \geq \phi$, we get

$$\begin{aligned} L'_i(\text{Alg}_0) &> y - 0.5 + \frac{\phi v_i - 1 - y}{\phi} = y\left(1 - \frac{1}{\phi}\right) - 0.5 - \frac{1}{\phi} + v_i \\ &\geq \phi - 1 - \frac{1}{\phi} + v_i - 0.5 = v_i - 0.5 \end{aligned}$$

► **Lemma 10.** *$L'_i(\text{Opt}) \leq L'_i(\text{Alg}_0)$ for all $i \in [m]$*

Proof. Towards a contradiction, let us assume that $L'_j(\text{Opt}) > L'_j(\text{Alg}_0)$ for some drone D_j . Observe that $L'_j(\text{Opt}) \leq L_j(\text{Opt}) \leq v_j$ as $T(\text{Opt}) = 1$. Using this and Lemma 8 we get,

$$L'_j(\text{Alg}_0) < L'_j(\text{Opt}) \leq v_j < \phi - 1 + L'_j(\text{Alg}_0)$$

As $L'_j(\text{Alg}_0)$ and $L'_j(\text{Opt})$ are both integer multiples of 0.5, the only value which satisfies the inequality is, $L'_j(\text{Opt}) = L'_j(\text{Alg}_0) + 0.5$. Let $L'_j(\text{Alg}_0) = k/2$ where k is an integer.

Case 1: $L'_j(\text{Alg}_0) = k/2$ is not an integer. This can only happen if D_j contains at least one medium job in Alg_0 (as small and large sized jobs have integral values and cannot add to give a non-integer). But then by Lemma 9, we get $L'_j(\text{Opt}) = L'_j(\text{Alg}_0) + 0.5 > v_j$, which is a contradiction as $L'_j(\text{Opt}) \leq v_j$.

Case 2: $L'_j(\text{Alg}_0) = k/2$ is an integer. This means that $L'_j(\text{Opt}) = (k + 1)/2$ is not an integer and hence, D_j has at least one medium job assigned to it in Opt . Therefore, by definition of the $R(\cdot)$ function, we get $L_j(\text{Opt}) \geq L'_j(\text{Opt}) + \phi - 1.5 = L'_j(\text{Alg}_0) + \phi - 1$. Using Lemma 8, we get $L_j(\text{Opt}) > v_j$, which is a contradiction as $L_j(\text{Opt}) \leq v_j$. ◀

Using Lemma 10, we get that

$$\sum_{i=1}^m L'_i(\text{Opt}) \leq \sum_{i=1}^m L'_i(\text{Alg}_0).$$

Observe that $\sum_{i=1}^m L'_i(\text{Opt}) = \sum_{i=1}^n R(\ell_i)$ and $\sum_{i=1}^m L'_i(\text{Alg}_0) = \sum_{i=1}^{n-1} R(\ell_i)$. Substituting this, we get

$$\sum_{i=1}^n R(\ell_i) \leq \sum_{i=1}^{n-1} R(\ell_i),$$

which is a contradiction. Therefore, our initial assumption must be wrong, implying that $T(\text{Alg}) \leq \phi$.

6.4 How much can the above analysis of LPT be improved?

Recall that the special case of DWP when all drones have the same battery life ($d_i = d_j \geq \max_k(\ell_k)$ for all $i, j \in [m]$) is equivalent to USP. It is known that LPT cannot give an approximation better than 1.54 for USP [28]. Therefore, the LPT algorithm can also not give an approximation ratio better than 1.54 for DWP.

7 Future work

A couple of immediate open problems are the following:

1. Can the implementation of the LPT heuristic be done in optimal time, i.e. $O((n + m) \cdot (\log m + \log n))$ time? We believe that it should be possible, since the jobs are known upfront, i.e., it is actually an offline problem.
2. Can the approximation ratio of the LPT heuristic be improved for DWP from ϕ ?

In general, delivering parcels from the warehouse using drones is a rich source of scheduling and vehicle routing problems. We mention two general directions:

1. As a concrete setting, consider a warehouse which has a truck and a drone, both of which operate independently. As in the paper, the goal is to assign parcels to the truck and the drone so that time taken to deliver is minimized. The parcels will be delivered by the drone using the same model as in this paper, whereas the truck will deliver using the traditional technique. A generalized version of the problem would involve multiple trucks and drones.
2. Some companies might have multiple warehouses and as such, a parcel can be delivered by any of the warehouses. As a concrete setting, consider the generalization of the DWP studied in this paper to the setting where there are *two* warehouses at different locations. The goal is to perform a two-level partition: first partition the parcels among the warehouses and then partition them among the drones. The goal is to deliver all the parcels as quickly as possible.

References

- 1 Pankaj K. Agarwal, Ravid Cohen, Dan Halperin, and Wolfgang Mulzer. Maintaining the union of unit discs under insertions with near-optimal overhead. In Gill Barequet and Yusu Wang, editors, *35th International Symposium on Computational Geometry, SoCG 2019, June 18-21, 2019, Portland, Oregon, USA*, volume 129 of *LIPICs*, pages 26:1–26:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPICs.SOCG.2019.26.
- 2 Niels A. H. Agatz, Paul C. Bouman, and Marie Schmidt. Optimization approaches for the traveling salesman problem with drone. *Transp. Sci.*, 52(4):965–981, 2018. doi:10.1287/TRSC.2017.0791.
- 3 Pasquale Ambrosio and Vincenzo Auletta. Deterministic monotone algorithms for scheduling on related machines. *Theor. Comput. Sci.*, 406(3):173–186, 2008. doi:10.1016/J.TCS.2008.06.050.
- 4 Nir Andelman, Yossi Azar, and Motti Sorani. Truthful approximation mechanisms for scheduling selfish related machines. *Theory Comput. Syst.*, 40(4):423–436, 2007. doi:10.1007/S00224-006-1316-9.
- 5 Vincenzo Auletta, Roberto De Prisco, Paolo Penna, and Giuseppe Persiano. Deterministic truthful approximation mechanisms for scheduling related machines. In Volker Diekert and Michel Habib, editors, *STACS 2004, 21st Annual Symposium on Theoretical Aspects of Computer Science, Montpellier, France, March 25-27, 2004, Proceedings*, volume 2996 of *Lecture Notes in Computer Science*, pages 608–619. Springer, 2004. doi:10.1007/978-3-540-24749-4_53.

- 6 François Bodin, Tristan Charrier, Arthur Queffelec, and François Schwarzentruber. Generating plans for cooperative connected uavs. In Jérôme Lang, editor, *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 5811–5813. ijcai.org, 2018. doi:10.24963/IJCAI.2018/846.
- 7 John L. Bruno, Edward G. Coffman Jr., and Ravi Sethi. Scheduling independent tasks to reduce mean finishing time. *Commun. ACM*, 17(7):382–387, 1974. doi:10.1145/361011.361064.
- 8 John Gunnar Carlsson and Siyuan Song. Coordinated logistics with a truck and a drone. *Manag. Sci.*, 64(9):4052–4069, 2018. doi:10.1287/MNSC.2017.2824.
- 9 Shushman Choudhury, Kiril Solovey, Mykel J. Kochenderfer, and Marco Pavone. Coordinated multi-agent pathfinding for drones and trucks over road networks. In *21st International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2022, Auckland, New Zealand, May 9-13, 2022*, pages 272–280. IFAAMAS, 2022. doi:10.5555/3535850.3535882.
- 10 Gregory Dobson. Scheduling independent tasks on uniform processors. *SIAM Journal on Computing*, 13(4):705–716, 1984. doi:10.1137/0213044.
- 11 Leah Epstein and Asaf Levin. Scheduling with processing set restrictions: Ptas results for several variants. *International Journal of Production Economics*, 133(2):586–595, 2011. Towards High Performance Manufacturing. doi:10.1016/j.ijpe.2011.04.024.
- 12 Donald K. Friesen. Tighter bounds for lpt scheduling on uniform processors. *SIAM Journal on Computing*, 16(3):554–560, 1987. doi:10.1137/0216037.
- 13 Michael R Garey and David S Johnson. *Computers and intractability*, volume 174. freeman San Francisco, 1979.
- 14 Michel Gendreau, Alain Hertz, and Gilbert Laporte. A tabu search heuristic for the vehicle routing problem. *Management Science*, 40(10):1276–1290, 1994.
- 15 Inge Li Gørtz, Marco Molinaro, Viswanath Nagarajan, and R. Ravi. Capacitated vehicle routing with non-uniform speeds. In Oktay Günlük and Gerhard J. Woeginger, editors, *Integer Programming and Combinatorial Optimization*, pages 235–247, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg. doi:10.1007/978-3-642-20807-2_19.
- 16 R. L. Graham. Bounds for certain multiprocessing anomalies. *Bell System Technical Journal*, 45(9):1563–1581, 1966.
- 17 R. L. Graham. Bounds on multiprocessing timing anomalies. *SIAM Journal on Applied Mathematics*, 17(2):416–429, 1969.
- 18 F. Guerriero, R. Surace, V. Loscrí, and E. Natalizio. A multi-objective approach for unmanned aerial vehicle routing problem with soft time windows constraints. *Applied Mathematical Modelling*, 38(3):839–852, 2014.
- 19 Erez Hartuv, Noa Agmon, and Sarit Kraus. Scheduling spare drones for persistent task performance under energy constraints. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2018, Stockholm, Sweden, July 10-15, 2018*, pages 532–540. International Foundation for Autonomous Agents and Multiagent Systems Richland, SC, USA / ACM, 2018. URL: <http://dl.acm.org/citation.cfm?id=3237463>.
- 20 Dorit S. Hochbaum and David B. Shmoys. Using dual approximation algorithms for scheduling problems theoretical and practical results. *J. ACM*, 34(1):144–162, 1987. doi:10.1145/7531.7535.
- 21 Dorit S. Hochbaum and David B. Shmoys. A polynomial approximation scheme for scheduling on uniform processors: Using the dual approximation approach. *SIAM J. Comput.*, 17(3):539–551, 1988. doi:10.1137/0217033.
- 22 Ellis Horowitz and Sartaj Sahni. Exact and approximate algorithms for scheduling nonidentical processors. *J. ACM*, 23(2):317–327, 1976. doi:10.1145/321941.321951.
- 23 Shlomo Karhi and Dvir Shabtay. Online scheduling of two job types on a set of multipurpose machines. *International Journal of Production Economics*, 150:155–162, 2014. doi:10.1016/j.ijpe.2013.12.015.

- 24 Jonghoe Kim and James R. Morrison. On the concerted design and scheduling of multiple resources for persistent uav operations. In *2013 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 942–951, 2013.
- 25 Jonghoe Kim, Byung Duk Song, and James R. Morrison. On the scheduling of systems of uavs and fuel service stations for long-term mission fulfillment. *J. Intell. Robot. Syst.*, 70(1-4):347–359, 2013. doi:10.1007/S10846-012-9727-0.
- 26 David Klaska, Antonín Kucera, and Vojtech Reháč. Adversarial patrolling with drones. In *Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems, AAMAS '20, Auckland, New Zealand, May 9-13, 2020*, pages 629–637, 2020. doi:10.5555/3398761.3398837.
- 27 Annamária Kovács. Fast monotone 3-approximation algorithm for scheduling related machines. In Gerth Støltting Brodal and Stefano Leonardi, editors, *Algorithms - ESA 2005, 13th Annual European Symposium, Palma de Mallorca, Spain, October 3-6, 2005, Proceedings*, volume 3669 of *Lecture Notes in Computer Science*, pages 616–627. Springer, 2005. doi:10.1007/11561071_55.
- 28 Annamária Kovács. New approximation bounds for lpt scheduling. *Algorithmica*, 57(2):413–433, 2010. doi:10.1007/S00453-008-9224-9.
- 29 Jan Karel Lenstra, David B. Shmoys, and Éva Tardos. Approximation algorithms for scheduling unrelated parallel machines. *Math. Program.*, 46:259–271, 1990. doi:10.1007/BF01585745.
- 30 Joseph Y.-T. Leung and Chung-Lun Li. Scheduling with processing set restrictions: A survey. *International Journal of Production Economics*, 116(2):251–262, 2008. doi:10.1016/j.ijpe.2008.09.003.
- 31 Joseph Y.-T. Leung and Chung-Lun Li. Scheduling with processing set restrictions: A literature update. *International Journal of Production Economics*, 175:1–11, 2016. doi:10.1016/j.ijpe.2014.09.038.
- 32 Joseph Y.-T. Leung and C. T. Ng. Fast approximation algorithms for uniform machine scheduling with processing set restrictions. *Eur. J. Oper. Res.*, 260(2):507–513, 2017. doi:10.1016/J.EJOR.2017.01.013.
- 33 Chung-Lun Li and Kangbok Lee. A note on scheduling jobs with equal processing times and inclusive processing set restrictions. *J. Oper. Res. Soc.*, 67(1):83–86, 2016. doi:10.1057/JORS.2015.56.
- 34 Chung-Lun Li and Qingying Li. Scheduling jobs with release dates, equal processing times, and inclusive processing set restrictions. *J. Oper. Res. Soc.*, 66(3):516–523, 2015. doi:10.1057/JORS.2014.22.
- 35 Chung-Lun Li and Xiuli Wang. Scheduling parallel machines with inclusive processing set restrictions and job release times. *Eur. J. Oper. Res.*, 200(3):702–710, 2010. doi:10.1016/J.EJOR.2009.02.011.
- 36 Amith Manoharan. Strategies for cooperative uavs using model predictive control. In Christian Bessiere, editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 5196–5197. ijcai.org, 2020. doi:10.24963/IJCAI.2020/738.
- 37 Gabriella Muratore, Ulrich M. Schwarz, and Gerhard J. Woeginger. Parallel machine scheduling with nested job assignment restrictions. *Operations Research Letters*, 38(1):47–50, 2010. doi:10.1016/j.orl.2009.09.010.
- 38 Chase C. Murray and Amanda G. Chu. The flying sidekick traveling salesman problem: Optimization of drone-assisted parcel delivery. *Transportation Research Part C: Emerging Technologies*, 54:86–109, 2015.
- 39 Ty Nguyen and Tsz-Chiu Au. Extending the range of delivery drones by exploratory learning of energy models. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2017, São Paulo, Brazil, May 8-12, 2017*, pages 1658–1660. ACM, 2017. URL: <http://dl.acm.org/citation.cfm?id=3091395>.

- 40 Marta Niccolini, Mario Innocenti, and Lorenzo Pollini. Multiple uav task assignment using descriptor functions. *IFAC Proceedings Volumes*, 43(15):93–98, 2010. 18th IFAC Symposium on Automatic Control in Aerospace.
- 41 Alena Otto, Niels A. H. Agatz, James F. Campbell, Bruce L. Golden, and Erwin Pesch. Optimization approaches for civil applications of unmanned aerial vehicles (uavs) or aerial drones: A survey. *Networks*, 72(4):411–458, 2018. doi:10.1002/NET.21818.
- 42 Jinwen Ou, Joseph Leung, and Chung-Lun Li. Scheduling parallel machines with inclusive processing set restrictions. *Naval Research Logistics (NRL)*, 55:328–338, June 2008. doi:10.1002/nav.20286.
- 43 Mark H. Overmars and Jan van Leeuwen. Maintenance of configurations in the plane. *J. Comput. Syst. Sci.*, 23(2):166–204, 1981. doi:10.1016/0022-0000(81)90012-X.
- 44 Sivakumar Rathinam and Raja Sengupta. Algorithms for routing problems involving uavs. In *Innovations in Intelligent Machines - 1*, volume 70 of *Studies in Computational Intelligence*, pages 147–172. Springer, 2007. doi:10.1007/978-3-540-72696-8_6.
- 45 Dvir Shabtay and Shlomo Karhi. Online scheduling of two job types on a set of multipurpose machines with unit processing times. *Computers and Operations Research*, 39(2):405–412, 2012. doi:10.1016/j.cor.2011.05.002.
- 46 Kaarthik Sundar and Sivakumar Rathinam. Algorithms for heterogeneous, multiple depot, multiple unmanned vehicle path planning problems. *J. Intell. Robotic Syst.*, 88(2-4):513–526, 2017. doi:10.1007/S10846-016-0458-5.
- 47 Paolo Toth and Daniele Vigo. *The Vehicle Routing Problem*. Society for Industrial and Applied Mathematics, 2002.
- 48 Feng Wu, Sarvapali D. Ramchurn, and Xiaoping Chen. Coordinating human-uav teams in disaster response. In Subbarao Kambhampati, editor, *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pages 524–530. IJCAI/AAAI Press, 2016. URL: <http://www.ijcai.org/Abstract/16/081>.
- 49 Miao Yu, Viswanath Nagarajan, and Siqian Shen. An approximation algorithm for vehicle routing with compatibility constraints. *Operations Research Letters*, 46(6):579–584, 2018. doi:10.1016/j.orl.2018.10.002.