Article

Maximizing the Average Environmental Benefit of a Fleet of Drones under a Periodic Schedule of Tasks

Vladimir Kats [1] and Eugene Levner [2]

[1]Institute for Industrial Mathematics, Ben Gurion University, Beer-Sheva, 8424902, Israel, vkats782@gmail.com; katzlu@bgu.ac.il

[2] School of Computer Science, Holon Institute of Technology, Holon, 5810201, Israel, levner@hit.ac.il

Abstract: Unmanned aerial vehicles (UAVs, drones) are not just a technological achievement based on modern ideas of artificial intelligence; they also provide a sustainable solution for green technologies in logistics, transport and material handling. In particular, using battery-powered UAVs to transport products can significantly decrease energy and fuel expenses, reduce environmental pollution, and improve the efficiency of clean technologies through improved energy-saving efficiency. We consider the problem of maximizing the average environmental benefit of a fleet of drones given a periodic schedule of tasks performed by the fleet of vehicles. To solve the problem efficiently, we formulate it as an optimization problem on an infinite periodic graph and reduce it to a special type of parametric assignment problem. We exactly solve the problem under consideration in $O(n^3)$ time, where n is the number of tasks performed by vehicles.

Keywords: sustainable scheduling; fleet of drones; maximizing the environmental benefit; polynomial time algorithm

1. Introduction

Transport by road and air is one of the largest contributors to the highest greenhouse gas emissions and fuel consumption in the logistics industry. In recent years, due to rising carbon dioxide emissions and fuel costs, the issue of sustainable daily scheduling of transport operations aimed at reducing environmental pollution, has attracted increasing interest and concern from large logistics companies. In such a situation, unmanned aerial vehicles (UAVs, drones) are not just a technological advancement based on modern ideas of artificial intelligence, but actually provide a sustainable solution to green technologies in logistics, transport and material handling In this light, the ability of UAVs to carry out sustainable autonomous deliveries efficiently and effectively has been researched and explored by large logistics companies (e.g., DHL Express, UPS), e-commerce retailers and companies (e.g., Walmart, Google) [1,2].

The environmental benefits of UAVs are universal and intertwined with economic and social benefits; their environmental friendliness has systematically improved in recent years as drone delivery technologies (e.g., batteries, autonomous navigation, obstacle avoidance and detection algorithms) have been greatly improved, and because drones are much smaller than trucks and airplanes, drones generally cost less and consume less energy and fuel per unit distance travelled. What's more, most delivery drones consume electricity, which can be generated from

clean energy sources such as solar or wind, so they emit fewer harmful emissions per unit of energy consumed compared to traditional trucks and airplanes, while also increasing the efficiency of clean technologies.

We observe that the environmental benefits of UAVs are mainly driven by the overall benefits and profits of modern digital and artificial intelligence-based technologies. They were comprehensively studied and classified by Dolgui and Ivanov [3,4]. Here, we select and highlight three main factors that directly lead to environmental benefits and economic profits:

- Using battery-powered UAVs in the air instead of trucks on the road can significantly reduce energy and fuel expenses,
- It reduces environmental pollution, carbon dioxide emissions and other negative impacts of transport processes on the environment,
- It improves the efficiency of clean technologies by increasing energy saving efficiency.

A broader and more detailed categorization of the relevant environmental benefits of UAVs is beyond the scope of this article. The interested reader is referred to the available comprehensive surveys and studies in this area cited therein [1,2, 5-7].

It is known that scheduling and routing of vehicles subject to external, in particular, environmental constraints, is a complex combinatorial problem, which, generally, is NP–hard, meaning it cannot be solved optimally for large instance sizes in reasonable (polynomial) computational time [8, 9]. In this regard, a practical alternative would be either to obtain efficient exact algorithms for practically important special cases of the problem, or to develop fast and sufficiently effective approximation algorithms. Increased theoretical and practical interest in this problem and its applications has been noted, for example, in a number of excellent reviews [1,2, 5-7]. In addition to the results presented in these surveys and the works cited therein, the main contribution of this work is as follows: we present and explore the problem of finding the optimal number of vehicles that maximizes the average UAV fleet profit per vehicle and thereby the fleet efficiency.

In what follows, we restrict our attention to the graph approach for maximizing the average fleet profit per vehicle and solve this optimization problem in strongly polynomial time, reducing it to a special type of parametric assignment problem.

For a detailed description of various problem formulations, models, and corresponding algorithms for the general drone fleet assignment problem, we refer the interested reader to the surveys [10-15], which provide excellent reviews of work in this area.

The rest of the article is organized as follows. In the next section we describe previous work. In Section 3 we describe the problem under study. In Section 4 we reformulate it as a graph-theoretic problem of maximizing the average total weight of a chain cover on an infinite graph; then we re-construct the infinite graph into an equivalent finite graph; as a result, the problem under consideration is reduced to the problem of covering the nodes of the last graph with a set of cycles that maximizes the average total weight. Section 5 reformulates the cycle-covering problem as a bi-matrix assignment problem and reduces the latter problem to a particular type of fractional assignment problem (FAP). Section 6 reduces FAP to a parametric assignment problem, and Section 7 solves it using Newton's method. Section 8 improves the complexity of the latter algorithm. Section 9 concludes the paper.

2. Previous work

Since we realize that the general problem under study is very complex (NP-hard in fact) and large in size, making it difficult to solve exactly, the challenge is to decompose it into smaller and simpler sub-problems that can be solved efficiently and thus provide the "building blocks" for solving the overall problem. We are interested in finding a special case that, on the one hand, makes the problem solvable, and, on the other hand, gives good upper/lower bounds on the objective functions of the general optimization problem.

The special case studied in this paper is an extension of the problem of minimizing the number of vehicles (airplanes, drones, robots, cars, etc.) needed to meet a fixed, periodically repeating schedule of tasks. This optimization problem has a long history in operations research. A non-periodic finite-horizon version of the problem, concerned with minimizing the number of tankers to meet a fixed schedule, was solved in 1954 by Dantzig and Fulkerson [16]. A few years later, Ford and Fulkerson [17] reduced that problem to finding a chain cover of minimum cardinality for finite partially ordered sets. Using an infinite periodic graph model and a chain covering, Karzanov and Livshits [18] elegantly reduced the periodic minimum-size vehicle problem to an assignment problem solvable in polynomial time. Kats [19] has modified and slightly simplified the Karzanov-Livshits algorithm. Orlin []20] proposed a chain covering algorithm for the periodic case where a finite number of tasks must be executed periodically over an infinite horizon and efficiently solved the problem as a finite network flow problem. Kats and Levner [21, 22] proposed a periodic graph model that solved a similar scheduling problem with non-Euclidean distances. Orlin [23] has solved the more general problem of minimizing the average fleet cost per day of flying subject to a fixed number of aircraft; however, this optimization problem is different from the problem in this study, and besides this, it involves a solution technique induced from dynamic minimum-cost network flows which is substantially different from the technique presented here. Campbell and Hardin [24] considered the problem of minimizing the number of vehicles required to make periodic deliveries to a set of customers, under the assumption that each delivery requires the use of a vehicle for a full day; they thoroughly examined the problem structure, evaluated its complexity, and presented an algorithm that optimally solved the problem for some special cases. Extensive reviews of UAV fleet size optimization techniques and industrial applications are presented, for example, in [12-15].

In this paper, we present and explore a more general particular case of the vehicle scheduling problem, namely, our goal is to find the optimal number of vehicles that maximizes the average environmental fleet benefit per vehicle and thereby the UAV fleet environmental efficiency. Note that the optimal number of vehicles in the latter problem may be strictly greater than the minimum number of vehicles required to meet a given periodic schedule. We propose a new fast algorithm the logic and main stages of which are presented in Figure 1.

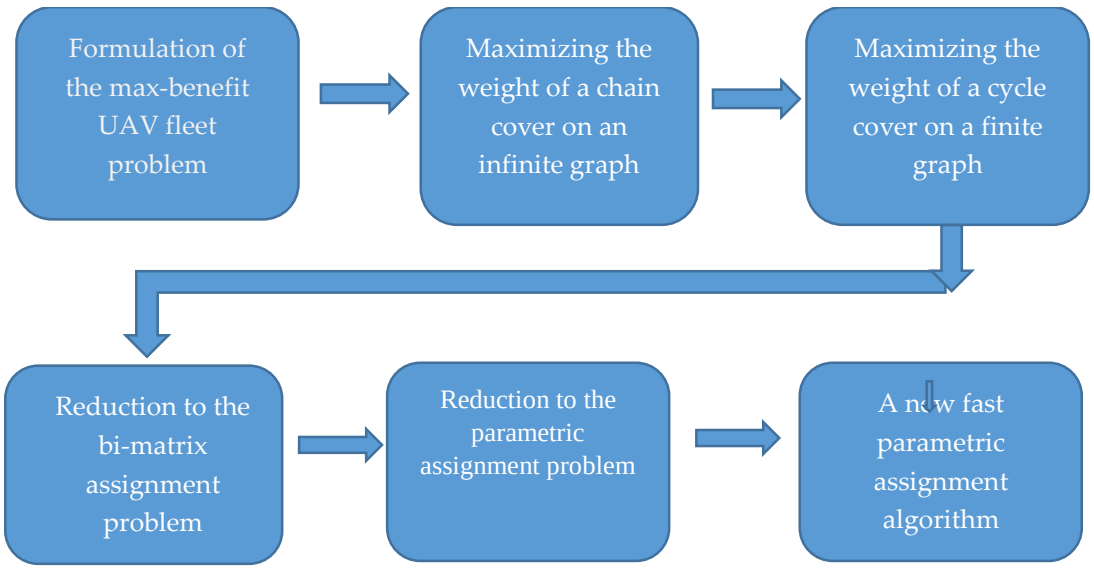

Figure 1. Schematic diagram of the proposed fast algorithm.

3. Description of the problem

Let $J_1, \ldots, J_n$ be a set of n tasks (e.g., flights) that must be carried out periodically by a fleet of drones, and let p denote a given period length. Associated with task J_i are non-negative real numbers a_i and b_i such that $a_i < p$ and $b_i < p$, where a_i and b_i are the start and finish time of task J_i in the time interval $[0, p)$, respectively. If $a_i < b_i$, then the kth iteration (or instance) of task J_i is executed in the time interval $[a_i + kp, b_i + kp)$. If $a_i > b_i$, then the kth iteration of task J_i is performed in the time interval $[a_i + kp, b_i + (k + 1)p)$, for $k = 0, 1, 2, \ldots$. Thus, the times a_i, b_i , i=1,…, n, define a predetermined periodic schedule for all tasks.

Any task can be done by any autonomous vehicle, and any vehicle can carry out any task. (We use this assumption to simplify the presentation of our algorithm; though, in fact, the proposed general graph approach can be extended to handle multiple aircraft types and solve efficiently other combinatorial problems). There is a known setup time r_{ij} between the sequential processing of instances of task J_i and task J_j by the same vehicle. The setup time r_{ij} is the required delay between the arrival of flight J_i and the departure of flight J_j, assuming the same aircraft is used for both flights. It is allowed that the arrival site s for flight J_i may differ from the departure site t for flight J_j; in this case, the setup time r_{ij} will include the deadhead time from s to t.

Unlike known models mentioned above, this study takes into account not only the size of the fleet, but simultaneously the environmental *benefit* (profit) accrued from using the UAV fleet, which depends on the fleet size and the assignment of tasks performed by the vehicles. Assume that profit e_{ij} is accrued from the use of an UAV when performing task J_j following task J_i; this quantity is known in advance, while the number of vehicles and the assignment of tasks to vehicles are the decision variables that need to be found.

Let K be the (unknown) number of vehicles required to meet periodic fixed schedule, and E be the total environmental profit obtained by all vehicles over period p. Then our problem to be solved is the following

Problem A. Assign tasks to vehicles so as to maximize the ratio

$$E/K = \frac{Total\ environmental\ benefit\ accrued\ from\ the\ use\ of\ the\ UAV\ fleet\ over\ period\ p}{Number\ of\ vehicles\ required\ to\ meet\ a\ fixed\ schedule\ of\ tasks},$$

that is, to maximize the average benefit obtained by the fleet of vehicles over the period, for a given task schedule, which is defined by the known times $\{a_i, b_i , i=1,\ldots. n\}$.

Remark 1. In the above verbal formulation of Problem **A**, we do not reveal the explicit dependence of the total environmental benefit on the distribution of vehicles among tasks. This will be explained and formalized below in Problems **D** and **FAP** in the following sections.

Remark 2. The reader may notice that the problem under consideration in its current formulation resembles the well-known vehicle allocation problem. The latter problem in its various forms is known to be NP-hard (see, for example, [8, 20, 25, 26]. However, in this paper we will prove that the problem under study can be solved in polynomial time; this is because this problem has a special structure that greatly reduces the problem complexity.

4. Reduction to graph problems

Consider an infinite weighted periodic graph G_∞ which is constructed as follows: $G_\infty = (N_\infty, A_\infty)$, where N_∞ is an infinite set of nodes and A_∞ is an infinite set of arcs. In this graph, node $i_k \in N_\infty$ represents the kth iteration of task J_i. A directed arc $(i_k, j_l) \in A_\infty$ leads from node i_k to node j_l iff the vehicle is able to carry out the lth iteration of task J_j after it has completed the kth iteration of task J_i. The weight of the arc (i_k, j_l) represents the profit e_{ij} collected by the vehicle from serving the task J_j following the task J_i. Let us formulate all profits e_{ij} in the form of the $n \times n$ matrix $\boldsymbol{E} = \|e_{ij}\|$.

Consider a periodic directed chain in the graph G_∞. Note that the period of such a chain can span several periods p defined in the previous section. The chain determines the schedule of an individual vehicle visiting a sequence of tasks, and the sum of its arc weights e_{ij} determines the profit earned by that vehicle during that period. The set of directed chains in G_∞ covering all nodes determines the total number of required vehicles, their schedule and the total environmental benefit obtained. Thus, the original allocation Problem **A** is reduced to the following graph-theoretic problem on the infinite graph G_∞:

Problem B. Find a periodic infinite chain cover of the graph G_∞ that maximizes the following average environmental benefit (profit) obtained by the UAV fleet:

$$E/K = \frac{Total\ environmental\ \ profit\ obtained\ by\ the\ UAV\ fleet\ in\ period\ p}{Number\ of\ vehicles\ required} =$$

$$= \frac{Sum\ of\ arc\ \ weights\ e_{ij}\ in\ chain\ cover\ in\ period\ p}{Number\ of\ directed\ chains\ covering\ nodes}.$$

The next step is to transform the infinite graph G_∞ into an equivalent finite graph. Due to the periodicity of the graph G_∞, we can roll up it into the so-called finite *generating graph*, $G_{gen} = (N_{gen}, A_{gen})$, defined as follows: node $i \in N_{gen}$ represents all periodic implementations of task J_i; in other words, all nodes $i_k \in N_\infty$, $0 \le k < \infty$, of the infinite graph G_∞ are "packed" into one node i of the graph G_{gen}. Similarly, all arcs $(i_k, j_l) \in A_\infty$ are "packed" into one arc $(i, j) \in A_{gen}$. Thus, every infinite, periodically repeated arc chain in the graph G_∞ can be transformed into a corresponding directed cycle in the graph G_{gen}. Knowing the generating graph G_{gen}, we can transform the problem **B** of covering the graph G_∞ with chains into the following problem **C** of covering the nodes of the graph G_{gen} with cycles. Let $\boldsymbol{c}$ denote a set of such cycles.

Problem C. Find a covering of the nodes of the graph G_{gen} with a set of cycles $\boldsymbol{c}$ so as to maximize the ratio

$$E/K = \frac{\sum_{(i,j)\in \boldsymbol{c}} e_{ij}}{Number\ of\ required\ vehicles}.$$

Next, we associate two weights with each arc $(i, j) \in A_{gen}$. The first weight is the profit e_{ij} introduced above. The second weight, denoted k_{ij}, is needed to calculate the required number of vehicles. It is defined similarly to model in [20]; namely, k_{ij} is the number of periods p that must exist between an arbitrary departure iteration $k(i)$ of a flight J_i and the departure iteration $l(j)$ of the flight J_j that is closest to J_i in the graph G_∞, provided that the two flights are operated by the same aircraft; it is possible that $j=i$. Then

$$k_{ij} = l(j) - k(i). \quad (1)$$

For reader's convenience, consider the definition of the weight k_{ij} in more detail. Let $k(i)$ be an arbitrary integer, and the symbol $k(i)$ denotes that the flight J_i occurs during the $k(i)$-th period. Further, suppose that in the graph G_∞ there are arcs from the node representing the flight J_i to all other nodes representing iterations (repetitions) of the flight J_j; it is clear that then, in our notation, the numbers $\{k(j)\}$ will denote the numbers showing in which periods those repetitions of J_j occur. Among these numbers $k(j)$, choose the minimum one – this number is denoted as $l(j)$. Then, as stated above, $k_{ij} = l(j) - k(i)$. In other words, after completing the $k(i)$th iteration of task J_i, the vehicle does not have enough time to arrive and carry out the $(l(j)-1)$th iteration of task J_j but has time to successfully perform the $l(j)$th iteration of task J_j.

Remark 3. Since the process under consideration is periodic, the found values k_{ij} are valid for all task iterations. As we will show shortly, the sum of k_{ij} over all cycles in any cycle-cover of the graph G_{gen} is equal to the number of vehicles required to meet a given schedule.

Remark 4. Formally, the main property of the parameter $l(j)$ can be presented as follows: Let d_i denote the duration of the flight J_i, that is,

$$d_i = b_i - a_i, \text{ if } a_i < b_i \text{ and}$$
$$d_i = b_i + p - a_i, \text{ if } a_i > b_i.$$

Then the following inequalities hold:

$$a_j + (l(j) - 1)\cdot p < a_i + k(i)\cdot p + d_i + r_{ij} \le a_j + l(j)\cdot p.$$

Thus, we introduce the $n\times n$ matrix $\boldsymbol{K} = \|k_{ij}\|$ with the entries k_{ij} just described. If the arc (i, j) does not exist in the graph G_{gen}, then we set $e_{ij} = 0$ and $k_{ij} = \infty$. To obtain this matrix, we need $O(n^2)$ time.

Consider a small example to illustrate the above definition of k_{ij} in Eq.(1). Assume that we have three tasks J_1, J_2, J_3; $p = 10$, $a(J_1) = a_1 = 5$, $b(J_1) = b_1 = 8$; $a(J_2) = a_2 = 9$, $b(J_2) = b_2 = 2$; $a(J_3) = a_3 = 4$, $b(J_3) = b_3 = 9$; $r_{12}=2$, $r_{13}=1$, $r_{21}=1$, $r_{23}=3$. Let us consider, for example, the $12th$ iteration of these tasks, i.e. set $k(1) =12$, $k(2) =12$ and $k(3) =12$; then the task J_1 will be executed in the time interval [125, 128], the task J_2 - in the time interval [129, 132] and the tasks J_3 in the time interval [124, 129]. Therefore, the nearest iteration of task J_2 that can be performed by the same vehicle, which has performed the 12^{th} iteration of J_1, will be the 13^{th} iteration. Respectively, the nearest iteration of task J_3, that can be performed by the same vehicle following the 12^{th} iteration of J_1,will be also the13^{th} iteration. Hence, $k_{12} = k_{13} =13 -12 =1$. Similarly, we obtain that $k_{23} = 14-12 =2$; $k_{21}= 1$.

At this point, let us recall a remarkable property of the weights k_{ij} in the periodic minimum-size vehicle problem, discovered by Karzanov and Livshits [18] in 1978 and independently by Orlin [20] in 1982, that establishes a direct connection between the *minimum number* of vehicles required and the *minimum-weight cycle-cover* of the graph G_{gen}, the arc weights of which are the weights k_{ij} defined by Eq. (1). Denote by $\boldsymbol{c}$ an arbitrary cycle cover of the generating graph G_{gen}, and $\boldsymbol{S}$ the set of all possible cycle covers of the graph G_{gen}.

The Karzanov-Livshits-Orlin Theorem. A *minimum-weight* cycle-cover, having the total weight $\min_{c\in S} \sum_{(i,j)\in c} k_{ij}$, provides the *minimum number* of vehicles K_{min} required to perform a given task schedule: $K_{min} = \min_{c\in S} \sum_{(i,j)\in c} k_{ij}$.

For our further analysis, we need to extend the above claim to formulate it for the bi-matrix optimization problem **C**:

Proposition 1. Let $\boldsymbol{c}$ be an arbitrary cycle cover of the generating graph G_{gen} with two arc weights, e_{ij} and k_{ij}, where k_{ij} is the weight defined by Eq. (1). Then the total weight $\sum_{(i,j)\in c} k_{ij}$ of the cycle-cover is equal to the number of vehicles K required to carry out all the tasks in the considered cycle-cover $\boldsymbol{c}$ in Problem **C**: $K = \sum_{(i,j)\in c} k_{ij}$.

Proof. Let σ be a simple cycle entering the cycle cover $\boldsymbol{c}$ of the generating graph G_{gen} with two arc weights, and let $s = \sum_{(i,j)\in\sigma} k_{ij}$. Consider the periodic graph G_∞ that corresponds to the initial problem **B**, which has generated the generating graph G_{gen}. Recall, that each node $i \in \sigma$ corresponds to an infinite number of nodes i_k in graph G_∞, $0 \le k < \infty$. As proven in [20], a cycle σ can be unpacked into exactly s chains in the graph G_∞ such that no node in one chain is linked to a node in another chain. From the description of problem **B** and the definition of the cycle weight, it follows that each chain corresponds to one vehicle performing the tasks of the chain and then the value $\sum_{(i,j)\in\sigma} k_{ij}$ is equal to the number of vehicles required to carry out all the tasks of the cycle σ. To complete the proof, it suffices to notice that the graph G_{gen} can be partitioned into a set $\boldsymbol{c}$ composed of a finite number of simple cycles and therefore the total number of chains $\sum_{(i,j)\in c} k_{ij}$ corresponding to all simple cycles in partition $\boldsymbol{c}$ is equal to the number of vehicles required, which proves the claim.

From Proposition 1 it follows that Problem **C** reduces to the following Problem **D**.

Problem D. Find a cycle-cover of the graph G_{gen} with a set of cycles $\boldsymbol{c}$, so as to maximize the ratio $E/K = \sum_{(i,j)\in c} e_{ij} / \sum_{(i,j)\in c} k_{ij}$.

The question remains: how to efficiently solve the resulting cycle-covering problem for the graph G_{gen}? We will answer this question in the following sections.

The illustrative example. For reader's convenience, consider a numerical example. It is adapted from the work of Orlin [20], who studied the problem of minimizing the number of aircraft to operate a fixed daily repeating set of flights; we generalize Orlin's example for the problem of maximizing the average fleet profit. While the latter numerical example was used in [20] to schedule daily aircraft flights, it is extended here to also serve to illustrate all the steps of the proposed algorithm for maximizing the average benefit of a drone fleet.

Model of the fixed daily-repeating set of drone flights.

Three daily flights are shown in Table 1.

Table 1. Daily required flights

Flight No.	Depart	Arrive
1	Honolulu 1:00 pm	Washington, DC 11:00 pm
2	New York 3:00 pm	Tokyo 4:00 am
3	London 1:00 pm	Paris 2:00 pm

It is also necessary to take into account the "deadhead" time, that is, the time that a given aircraft takes, after completing a flight, to reach the departure location of the next scheduled flight. The deadhead times are given in Table 2.

Table 2. The deadhead flight times in hours

	Honolulu	London	New york
Paris	15	1	7
Tokyo	8	12	13
Washington	10	7	1

In the notation, introduced above, the input data from Tables 1 and 2 are given in Table 3. Tasks J_1, J_2, and J_3 represent flights 1, 2 and 3, respectively.

Table 3. Input data

Flight No.	Departure	Arrive		Set-up time			
i	a_i	b_i		r_{ij}	1	2	3
1	13	23		1	10	1	7
2	15	4		2	8	13	12
3	13	14		3	15	7	1

We consider a daily-repeating schedule of flights, therefore, period length p is 24 hours.

Reduction to a problem on a periodic graph

Let the profits be given by the following matrix $\boldsymbol{E}$ shown in Table 4.

Table 4. Matrix $\boldsymbol{E}$

$$E = \begin{Vmatrix} 300 & 900 & 600 \\ 600 & 600 & 1200 \\ 900 & 300 & 300 \end{Vmatrix}$$

Using the flight schedule given in Table 3, it is possible to plot the graph G_∞. A fragment of the graph G_∞ drawn for three consecutive periods, starting with period 1, is presented in Fig. 2.

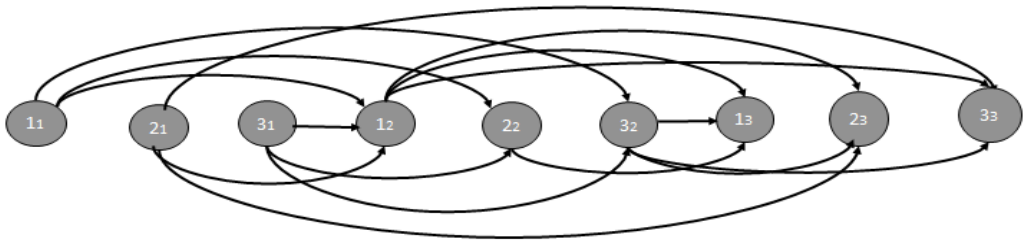

Figure 2. A 3-period fragment of the graph G_∞.

The nodes in Fig. 2 depict the daily flights repeated for three periods; node i_k depicts flight i (i=1,2,3) during period k (k=1,2,3). Fig. 2 shows only the arcs (i_k, j_l) between the nearest iterations k and l of tasks J_i and J_j. Each arc (i_k, j_l) in G_∞ has the associated weight $e(i_k, j_l)$, however, it is not shown in Fig. 2 in order not to overload it. At this point, our initial fleet assignment problem is equivalent to finding the optimal number of infinite periodic disjoint paths covering all nodes and having maximum average profit per path per period.

Construction of the generating graph

According to the above description, the graph G_{gen} can be portrayed as follows (see Fig. 3):

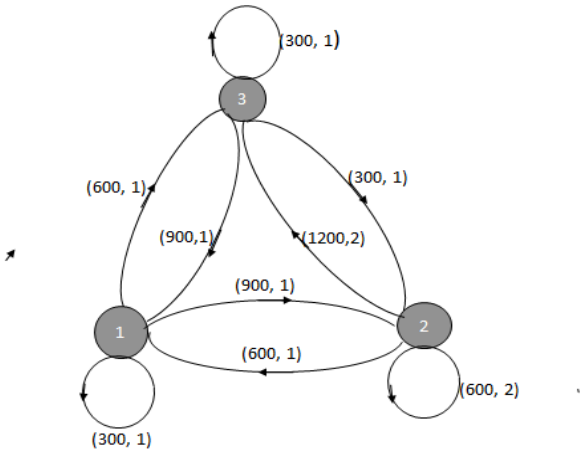

Figure 3. Generating graph G_{gen}.

Each arc is double-weighted, and the weight of arc (i, j) has the form (e_{ij}, k_{ij}), where the number k_{ij} is determined by arc (i_k, j_l) in Fig. 2, $k_{ij} = l - k$, $(i, j = 1, 2, 3)$. Note that the generating network for the airplane scheduling example presented in [20] is different from the graph G_{gen}; namely, the arcs of the graph G_{gen} are double-weighted, whereas in the Orlin graph the weights on arcs (i, j) are specified by a single number.

Calculation of the second weight k_{ij}.

According to the definition of the second weight k_{ij} and the data in Table 1, we see that flights 1 and 3 can be repeated every period (every day) by the same aircraft (the arc from node 1_1 to node 1_2 and the arc from node 3_1 to node 3_2 in Fig. 2); therefore, $k_{11}=k_{33}=1$. Flight 2 can be

repeated by the same vehicle every two days; therefore, k_{22} =2. Further, if an aircraft starts its flight no. 2 at 15:00 in the first period (node 2_1 in Fig. 2), then it can start the nearest flight no. 3 only after 13+12=25 hours, i.e. on the third day at 13:00 (node 3_3 in Fig. 2); hence, $k_{23} = 3 - 1 = 2$. Similarly, we compute all other values k_{ij} shown in Fig. 3 and in Table 5.

Table 5**.** Matrix ***K***

	1	1	1
K =	1	2	2
	1	1	1

The illustrative numerical example will be continued and completed in Section 7.

5. Reduction to the fractional assignment problem

In this section, we reformulate Problem **D** as a fractional assignment problem. Recall that an *assignment* can be stated in the form of $n \times n$ matrix A whose entries, denoted a_{ij}, are 0 or 1, and a_{ij} =1 occurs in each row and each column of the matrix A exactly once.

Consider Problem **D**, defined in the previous section, and the corresponding cycle cover of the graph G_{gen}**.**

Proposition 2. The set of all cycles in the cycle cover $\boldsymbol{c}$= (C_1, C_2, …, C_q) of the graph G_{gen}, where q is the number of cycles in $\boldsymbol{c}$, can be considered as an $n \times n$ assignment A in the matrix form defined as follows:

a_{ij} = 1 if and only if the arc (i,j) from G_{gen} belongs to $\boldsymbol{c}$, and a_{ij} = 0, otherwise. (2)

Indeed, from the definition of the cycle-cover it follows that the matrix A contains n rows and n columns, its elements are 1 or 0, and each a_{ij} =1 occurs exactly once for each row i and each column j (i, j =1,…, n), which proves the claim.

Given a profit matrix $\boldsymbol{E}$, a weight matrix $\boldsymbol{K}$ and an arbitrary assignment matrix A, the profit and weight values corresponding to the assignment A are defined, respectively, as follows:

$$E(A) = \sum_{(i,j)\in A} e_{ij}a_{ij} \text{ and } K(A) = \sum_{(i,j)\in A} k_{ij}a_{ij}.$$

In the above expression and wherever necessary, the notation $(i,j)\in A$ denotes all pairs of indices i, j corresponding to entries a_{ij} = 1 of the assignment matrix A. For ease of notation, in the expressions $E(A)$ and $K(A)$ below we will omit the symbols a_{ij}. In what follows, in accordance with (2), instead of maximizing $P = \sum_{(i,j)\in c} e_{ij} / \sum_{(i,j)\in c} k_{ij}$ over all cycle covers of the graph G_{gen} (Problem **D**), we will focus on finding the *optimal assignment* that maximizes $P = E(A)/K(A) = \sum_{(i,j)\in A} e_{ij} / \sum_{(i,j)\in A} k_{ij}$.

Let us denote by **C** the set of all assignments A in the form of $n \times n$ matrices. Now Problem **D** in section 2 reduces to the following fractional assignment problem (**FAP**):

Problem FAP. Given two $n \times n$ matrices $\boldsymbol{E}$ and $\boldsymbol{K}$, find the *optimal* assignment A^*, common to the matrices $\boldsymbol{E}$ and $\boldsymbol{K}$, that is, the one that maximizes the average fleet profit per vehicle $P = E(A)/K(A)$:

$$P^* = E(A^*)/\, K(A^*) = \max_{A\in C} E(A)/K(A) = \max_{A\in C} \sum_{(i,j)\in A} e_{ij} \,/ \sum_{(i,j)\in A} k_{ij}.$$

The following statement shows that the elements of matrix $\boldsymbol{K}$ in the considered problem are the small positive integers not exceeding 3; in fact, this is an important property that reduces the complexity of the problem by a factor of n.

Proposition 3. If the arc $(i, j) \in G_{gen}$ then, for any element k_{ij} of matrix $\boldsymbol{K}$, it holds that $k_{ij} \le 3$.

Proof. Consider the $k(i)$th iteration of task J_i that starts at time $a_i + k(i)\cdot p$, and let $l(j)$ be the nearest iteration of task J_j that the vehicle can perform after the $k(i)$th iteration of task J_i. Consider two cases, $a_i < b_i$ and $a_i > b_i$.

(i) $a_i < b_i$.

In this case, the $k(i)$th iteration of task J_i ends at time $b_i + k(i)\cdot p$. Then the nearest iteration of task J_j satisfies the following inequalities:

$$a_j + [l(j) - 1]\cdot p < b_i + k(i)\cdot p + r_{ij} \le a_j + l(j)\cdot p,$$

or

$$a_j + [l(j) - k(i) - 1]\cdot p < b_i + r_{ij} \le a_j + [l(j) - k(i)]\cdot p.$$

(ii) $a_i > b_i$.

In this case, the $k(i)$th iteration of task J_i ends at time $b_i + [k(i) + 1]\cdot p$. Then the nearest iteration of task J_j satisfies the following inequalities:

$$a_j + [l(j) - 1]\cdot p < b_i + [k(i) + 1]\cdot p + r_{ij} \le a_j + l(j)\cdot p,$$

or

$$a_j + [l(j) - k(i) - 1]\cdot p < b_i + p + r_{ij} \le a_j + [l(j) - k(i)]\cdot p.$$

From the definition of $k_{ij} = l(j) - k(i)$ in Section 2, it immediately follows that each element k_{ij} is an integer satisfying the following relations:

$$a_j + (k_{ij} - 1)\cdot p < b_i + r_{ij} \le a_j + k_{ij}\cdot p, \text{ if } a_i < b_i$$

and

$$a_j + (k_{ij} - 1)\cdot p < b_i + p + r_{ij} \le a_j + k_{ij}\cdot p, \text{ if } a_i > b_i.$$

It means that, due to the given time constraints, the element k_{ij} is equal to the minimum number of periods p that the vehicle must skip before it can start task J_j after it have started task J_i. Since each task (flight) J_j is to be executed in every period p, the vehicle must skip k_{ij} instances of task J_j in the skipped periods.

It is natural to assume that all the setup times $r_{ij} < p$; then, from the definition of k_{ij}, we derive that the largest value of k_{ij} can appear only in the following inequalities

$$a_j + (k_{ij} - 1)\cdot p < b_i + p + r_{ij} \le a_j + k_{ij}\cdot p. \quad (3)$$

Suppose that $b_i > a_j$ and $p > r_{ij} > p - (b_i - a_j)$. Then, on the one hand, we have that

$$b_i + p + r_{ij} > b_i + p + p - (b_i - a_j) = a_j + 2p, \quad (4)$$

and, on the other hand, $b_i < a_j + p$; then we have that

$$b_i + p + r_{ij} \le (a_j + p) + p + p = a_j + 3p. \quad (5)$$

Comparing inequalities (3), (4) and (5), we find that in this case, $k_{ij} = 3$. Q.E.D.

6. Reduction to the parametric assignment problem

In this section, we reduce the fractional assignment problem **FAP** with the objective function P defined above to a parametric assignment problem. Let us introduce a parameter λ, which for each assignment A from the set **C** of all assignments satisfies the following inequality:

$$\sum_{(i,j)\in A} e_{ij} / \sum_{(i,j)\in A} k_{ij} \le \lambda, \text{ for all } A \in \mathbf{C}. \quad (6)$$

Since the sum $\sum_{(i,j)\in A} k_{ij}$ is positive in any assignment A, inequalities (6) can be rewritten as follows:

$$\sum_{(i,j)\in A} e_{ij} \le \lambda \sum_{(i,j)\in A} k_{ij}$$

or

$$0 \le \lambda \sum_{(i,j)\in A} k_{ij} - \sum_{(i,j)\in A} e_{ij}$$

and, finally,

$$0 \le \sum_{(i,j)\in A} (\lambda\, k_{ij} - e_{ij}) \text{ for all } A \in \mathbf{C} \quad (7)$$

Thus, Problem **FAP** of finding the maximum average profit P^* presented in the previous section takes the following form:

To find the minimum λ that satisfies (6), or, equivalently, to find the minimum λ that satisfies (7).

Consider the matrix $\boldsymbol{W}(\lambda) = \lambda \cdot \boldsymbol{K} - \boldsymbol{E}$ with entries $w_{ij} = (\lambda \cdot k_{ij} - e_{ij})$, called *arc costs*, or simply *costs*. Consider some fixed value of λ. Let $A^*(\lambda)$ denote the assignment with the minimum cost across all the assignments $A \in \mathbf{C}$, defined as follows:

$$\sum_{(i,j)\in A^*(\lambda)} (\lambda \cdot k_{ij} - e_{ij}) = \min_{A\in \mathbf{C}} \sum_{(i,j)\in A} (\lambda \cdot k_{ij} - e_{ij}), \quad (8)$$

and denote the latter function, called the *minimum-cost function*, by $L(\lambda)$:

$$L(\lambda) = \sum_{(i,j)\in A^*(\lambda)} (\lambda \cdot k_{ij} - e_{ij}). \quad (9)$$

Proposition 4 below states that the assignment problem of finding the maximum average fleet profit P^*, defined in Problem **FAP,** can be reduced to the following parametric minimum-cost assignment problem **PAP**:

Problem PAP. Find the value of the parameter $\lambda = \lambda^*$ for which the minimum-cost function $L(\lambda)$ is equal to zero:

$$L(\lambda) = \sum_{(i,j)\in A^*(\lambda)} (\lambda \cdot k_{ij} - e_{ij}) = 0, \quad (10)$$

and, further, together with the optimal value of the parameter λ^*, find the corresponding minimum-cost assignment $A^*(\lambda^*)$ for the matrix $\boldsymbol{W}(\lambda^*)$; for the simplicity of notation we denote the latter assignment by Φ^*: $\Phi^* = A^*(\lambda^*)$.

Proposition 4. Let λ^* be the optimal solution to the problem **PAP**, and $\Phi^* = A^*(\lambda^*)$ be the corresponding minimum-cost assignment for the matrix $\boldsymbol{W}(\lambda^*)$. Then

$$\lambda^* = P^* = \sum_{(i,j)\in A^*} e_{ij} / \sum_{(i,j)\in A^*} k_{ij} = \max_{A\in \mathbf{C}} \sum_{(i,j)\in A} e_{ij} / \sum_{(i,j)\in A} k_{ij},$$

$$\text{and } \Phi^* = A^*,$$

that is, in meaningful terms, (i) the value of $\lambda = \lambda^*$ defined by expression (10) is equal to the optimal profit value P^*, which we are looking for, and (ii) the assignment Φ^*, which is optimal for the problem **PAP,** coincides with the optimal assignment A^* for the maximum average fleet profit in the problem **FAP**.

Proof. From equation (10), we have:

$$L(\lambda^*) = \sum_{(i,j)\in A^*(\lambda^*)} (\lambda^* \cdot k_{ij} - e_{ij}) = 0. \quad (11)$$

In addition, from (8), we have:

$$\sum_{(i,j)\in A^*(\lambda^*)} (\lambda^* \cdot k_{ij} - e_{ij}) = \min_{A\in \mathbf{C}} \sum_{(i,j)\in A} (\lambda^* \cdot k_{ij} - e_{ij}).$$

Then

$$\sum_{(i,j)\in A} (\lambda^* \cdot k_{ij} - e_{ij}) \ge 0 \text{ for all } A \in \mathbf{C}. \quad (12)$$

Thus, from (11), we obtain that

$$\lambda^* = \sum_{(i,j)\in A^*(\lambda^*)} e_{ij} / \sum_{(i,j)\in A^*(\lambda^*)} k_{ij},$$

and from (12), we have

$$\lambda^* \geq \sum_{(i,j)\in A} e_{ij} / \sum_{(i,j)\in A} k_{ij} \text{ for all } A \in \mathbf{C},$$

that is, $\lambda^* = P^*$ and $\Phi^* = A^*$, which completes the proof.

The following claim is an extension of Proposition 4 that formulates a way of how we can optimally and efficiently solve the optimization problem PAP:

Corollary. Let λ^* be the optimal parameter value, i.e., such that

$$L(\lambda^*) = \sum_{(i,j)\in A^*(\lambda)} (\lambda \cdot k_{ij} - e_{ij}) = 0.$$

Then the parameter λ^* represents the maximum average profit per vehicle in the problem under study, and $K(A^*) = \sum_{(i,j)\in A^*} k_{ij}$ is equal to the optimal number of required vehicles, where, $A^* = A^*(\lambda^*)$.

In the following sections, we propose two strongly polynomial time algorithms for the parametric assignment problem under consideration, which in turn optimally solve the problem of maximizing the average profit of a fleet of identical vehicles.

7. A Newton-type algorithm for the parametric assignment problem

Consider the cost value $w(A, \lambda) = \sum_{(i,j)\in A} (\lambda \cdot k_{ij} - e_{ij})$ of an arbitrary assignment A, whose costs are the elements of the matrix $\boldsymbol{W}(\lambda)$. This value is a linear function $w(A, \lambda) = (d_A \lambda - f_A)$ of the parameter λ, where $d_A = \sum_{(i,j)\in A} k_{ij}$ is the slope and $f_A = \sum_{(i,j)\in A} e_{ij}$. In the UAV model considered, each $k_{ij} \leq 3$, so $d_A \leq 3n$. Let A be a collection of q cycles: $A = (C_1, C_2, \ldots, C_q)$. Consider an arbitrary cycle $C \in (C_1, C_2, \ldots, C_q)$. The sum of the elements k_{ij} over the cycle C is not less than one, $\sum_{(i,j)\in C} k_{ij} \geq 1$. Indeed, if $C = \{i_1, i_2 = j(i_1), i_3 = j(i_2), \ldots, i_s = j(i_{s-1}), i_1 = j(i_s)\}$, then after task J_{i1}, the vehicle performs task J_{i2}, and after task J_{is}, the vehicle again performs task J_{i1}, but not at the same iteration as at the previous one. Two iterations of task J_{i1}, carried out by the same vehicle, are separated by a time interval equal to $\sum_{(i,j)\in C} k_{ij} \geq 1$ periods. Therefore, $1 \leq d_A \leq 3n$.

There is a set of linear functions corresponding to different assignments A of the matrix $\boldsymbol{W}(\lambda)$, each of which has the form:

$$\sum_{(i,j)\in A} (\lambda \cdot k_{ij} - e_{ij}),\ A \in \mathbf{C}.$$

According to (8) and (9), $L(\lambda)$ is a lower bound for the costs of all assignments $A \in \mathbf{C}$; it is an increasing concave piecewise linear function of the parameter λ with slope $\sum_{(i,j)\in A^*(\lambda)} k_{ij}$. Since $1 \leq \sum_{(i,j)\in A^*(\lambda)} k_{ij} \leq 3n$, the number of pieces in $L(\lambda)$ does not exceed $3n$.

Let us select some arbitrary value $\lambda = \lambda'$. Determine the corresponding minimum-cost assignment $A' = A^*(\lambda')$ and the corresponding cost

$$L(\lambda') = \sum_{(i,j)\in A'} (\lambda' k_{ij} - e_{ij}). \quad (13)$$

Consider the function $d_{A'} \lambda - f_{A'} = \sum_{(i,j)\in A'} (\lambda \cdot k_{ij} - e_{ij})$. Let us select a starting value λ' such that it is so small that $L(\lambda') < 0$, then $\lambda' < \sum_{(i,j)\in A'} e_{ij} / \sum_{(i,j)\in A'} k_{ij} \leq \lambda^*$. The first inequality follows from (13); the second inequality follows from (12). Now we can set a new value of λ' that is equal to $\sum_{(i,j)\in A'} e_{ij} / \sum_{(i,j)\in A'} k_{ij}$. Then determine the minimum-cost assignment for the updated $\lambda = \lambda'$. This procedure must be continued until the optimal solution λ^* of the problem **PAP** is found. The following algorithm **A1** implements the described idea.

7.1 Algorithm A1

Initialization

Step 1. Solve the standard assignment problem for the known matrix $\boldsymbol{K}$ (using a standard assignment algorithm). Denote by I the obtained optimal (minimum-cost) assignment for this matrix.

Step 2. Calculate the average profit λ_0 received for the obtained assignment I:

$$\lambda_0 = \sum_{(i,j)\in I} e_{ij} / \sum_{(i,j)\in I} k_{ij}.$$

(Note that at this stage $\lambda_0 \le \lambda^*$, where λ^* is the maximum average profit that we are looking for).

Step 3. Set $i = 0$.

Iterative procedure

Step 4. Find the minimum-cost assignment $A^*(\lambda_i)$ for the matrix $\boldsymbol{W}(\lambda_i)$.

Step 5. Calculate the average profit λ_{i+1} of the assignment $A^*(\lambda_i)$:

$$\lambda_{i+1} = \sum_{(i,j) \in A^*(\lambda i)} e_{ij} / \sum_{(i,j) \in A^*(\lambda i)} k_{ij}.$$

Step 6. If $\lambda_{i+1} > \lambda_i$ then {set $i = i + 1$; go to Step 4},
else go to step 7.

Solution

Step 7. Set the maximum average profit per vehicle: $P^* = \lambda^* = \lambda_i$.
Set the optimal assignment $\Phi^* = A^*(\lambda_i)$

Determine the optimal number of vehicles needed to meet the obtained schedule:

$$K = \sum_{(i,j) \in \Phi^*} k_{ij}.$$

End.

The proposed algorithm can be considered as a discrete version of Newton's optimization method adapted to solving equation (10).

7.2 Complexity of Algorithm A1

In Algorithm **A1**, in the iterative procedure (Steps 4-6 above) the slope of the minimum-cost function $L(\lambda)$in (9), $\sum_{(i,j) \in A^*(\lambda i)} k_{ij}$, decreases, i.e.

$$\sum_{(i,j) \in A^*(\lambda i)} k_{ij} > \sum_{(i,j) \in A^*(\lambda i+1)} k_{ij}.$$

Therefore, steps 4-6 return at most $O(n)$ times. Each pass of the iterative procedure solves once the assignment problem. The complexity of solving the assignment problem is $O(n^3)$, which is repeated at most $O(n)$ times. Thus, the overall complexity of algorithm **A1** is $O(n^4)$.

The illustrative example (continued).

Reduction to the fractional FAP and parametric PAP problems.

Problem FAP. Given two $n \times n$ matrices $\boldsymbol{E}$ and $\boldsymbol{K}$, defined at the previous step, find the *optimal* assignment A^*, common to the matrices $\boldsymbol{E}$ and $\boldsymbol{K}$, that is, the one that maximizes the average fleet profit per vehicle $P = E(A)/K(A)$:

$$P^* = E(A^*)/\, K(A^*) = \max_{A \in C} E(A)/K(A) = \max_{A \in C} \sum_{(i,j) \in A} e_{ij} \,/ \sum_{(i,j) \in A} k_{ij}.$$

To solve this problem, we reduce it to the following parametric assignment problem:

Problem PAP. Find the value of the parameter $\lambda = \lambda^*$ for which the minimum-cost function $L(\lambda)$ is equal to zero: $L(\lambda) = \sum_{(i,j) \in A^*(\lambda)} (\lambda \cdot k_{ij} - e_{ij}) = 0$, and find the corresponding minimum-cost assignment $A^*(\lambda^*)$ for the matrix $\boldsymbol{W}(\lambda^*)$.

Solution of the parametric assignment problem by the Newton-type algorithm.

Let us solve the standard minimum-weight assignment problem with the matrix $\boldsymbol{K}$. We can see that assignment $I = \{C_1 = [(1, 2), (2, 1)]; C_2 = (3, 3)\}$ is optimal for this standard single-matrix problem, $\sum_{(i,j) \in I} k_{ij} = k_{1,2} + k_{2,1} + k_{3,3} = 3$. This is the minimum number of vehicles required to meet the given flight schedule.

Calculate the average profit received with the assignment I:

$$\lambda_0 = (900 + 600 + 300)/(1 + 1 + 1) = 600.$$

The value λ_0 can be taken as the lower bound of the desired maximum average profit P^*, $600 \le P^*$.

Consider the matrix $\boldsymbol{W}(\lambda_0 = 600)$ presented in Table 6:

Table 6. Matrix $\boldsymbol{W}(\lambda_0 = 600)$.

	600·1 - 300	600·1 - 900	600·1 - 600		300	-300	0
$W(600) =$	600·1 - 600	600·2 - 600	600·2 -1200	=	0	600	0
	600·1 -900	600·1 - 300	600·1 - 300		-300	300	300

The optimal assignment $A^*(\lambda_0 = 600)$ of the matrix $\boldsymbol{W}(\lambda_0 = 600)$ is:

$$A^*(\lambda_0 = 600) = \{C = [(1, 2), (2, 3), (3, 1)]\}.$$

The cost of the assignment $A^*(\lambda_0 = 600)$ at the point $\lambda_0 = 600$ is negative, $w(A^*, \lambda_0) = -300 + 0 + (-300) = -600$. Therefore, $\lambda_0 = 600$ is not an optimal solution to the **PAP** problem considered. This means that the average profit λ_1 obtained with the assignment $A^*(\lambda_0)$ is greater than λ_0: $\lambda_1 = (900 + 1200 + 900)/(1 + 2 + 1) = 750$

Now, in a similar way, consider the matrix $\boldsymbol{W}(\lambda_1 = 750)$ presented in Table 7:

Table 7. Matrix $\boldsymbol{W}(\lambda_1 = 750)$

	750·1 - 300	750·1 - 900	750·1 - 600		450	-150	150
$W(750) =$	750·1 - 600	750·2 - 600	750·2 -1200	=	150	900	300
	750·1 -900	750·1 - 300	750·1 - 300		-150	450	450

The optimal assignment $A^*(\lambda_1 = 750)$ of the matrix $\boldsymbol{W}(\lambda_1 = 750)$ is the same as $A^*(\lambda_0 = 600)$. The cost of the assignment $A^*(\lambda_1 = 750)$ at the point $\lambda_1 = 750$ is zero, $w(A^*, \lambda_1) = -150 + 300 + (-150) = 0$. Thus, $\lambda^* = \lambda_1 = 750$ is the optimal solution to our PAP problem, and according to , the maximum average profit per aircraft is $P^* = \lambda^* = 750$.

Solution of the original vehicle fleet optimization problem.

The optimal assignment $A^*(\lambda_1 = 750)$ consists of only one cycle $C = [(1,2), (2,3), (3,1)]$. It corresponds to the following optimal sequence of flights:
Flight 1 from Honolulu to Washington DC, then deadheading flight to New York, then
Flight 2 from New York to Tokyo, then deadheading flight to London, then
Flight 3 from London to Paris, then deadheading flight to Honolulu.

The considered circular route C takes four days, $\sum_{(i,j) \in C} k_{ij} = k_{1,2} + k_{2,3} + k_{3,1} = 4$. This schedule requires four aircraft, each repeating the same sequence of flights as the previous one, lagging behind it in time by a day. This is the optimal number of the aircraft in the considered example. Note that in this illustrative example, the optimal number of vehicles is four, and the minimum number of vehicles required to meet a given flight schedule is three; the optimal average fuel efficiency obtained in this example (i.e. for 4 vehicles) is 20 % better than the corresponding fuel efficiency for the minimum number of vehicles required: $(750 - 600)/750 = 0.2$.

Discussion. The Newton-type algorithm proposed in this section is relatively simple and easy to program. The question arises whether an algorithm of a similar or even better complexity can be obtained for the problem under study if one will exploit the existing polynomial algorithms for the general fractional assignment problem. Unfortunately, all polynomial algorithms for the general fractional assignment problem known to us (see, e.g., [26-30]) have the total running time similar or worse than $O(n^4)$, even in the case where the coefficients of the linear function in the denominator of the fractional objective are restricted to the values {0, 1}. However, unlike these studies, we found another way to speed up the solution time of the

Newton type algorithm. In the next section, we present an improved algorithm that can solve the parametric assignment problem under study in $O(n^3)$ time, and, hence, can optimally solve the original max-benefit UAV fleet problem in $O(n^3)$ time .

8. A faster parametric assignment algorithm

8.1 Comparison of two parametric assignment problems

Reduction of the original aircraft profit maximization problem **A** to the special-type parametric assignment problem **PAP** formulated in Section 6 opens an opportunity to exploit another solution algorithm which is faster than the Newton-type algorithm proposed in the previous section. For this purpose, we adapt and use, after appropriate adaptation, a fast and elegant parametric assignment algorithm developed more than a decade ago by Elizabeth Gassner and Bettina Klinz [31].

Since the parametric assignment problem solved by Gassner and Klinz is somewhat different from our assignment problem PAP, we should make necessary changes to the Gassner-Klinz algorithm (GKA) to make the adapted version of GKA applicable to solve the PAP under consideration. We begin by comparing two related parametric assignment problems, focusing on their differences (see Table 8). Note that the significant difference between the two studies is that the present work is motivated and focused on the practical aircraft fleet assignment problem.

Table 8. Comparison of two assignment problems

	Gassner & Klinz [31]	Problem **PAP** in this paper
Problem formulation	Given a bipartite graph G and parametric arc costs $c_\lambda(i, j) = (c_{ij} - \lambda \cdot b_{ij})$, find the *minimum of objective function* $z(\lambda) = \{\sum_{(i,j)\in A} c_\lambda(i,j)$: A is an assignment in $G\}$, *for all* $\lambda \in R$ together with the corresponding optimal assignments	Given a matrix $\boldsymbol{W}$ with parametric entry costs $w_\lambda(i, j) = (\lambda \cdot k_{ij} - e_{ij})$ and the *minimum cost function* $L(\lambda)$ defined as $L(\lambda) = \sum_{(i,j)\in A^*(\lambda)} (\lambda \cdot k_{ij} - e_{ij})$, find a *parameter value* $\lambda = \lambda^*$ for which the $L(\lambda) = 0$ and the corresponding optimal assignment $A^*(\lambda^*)$ (see Fig. 4)
Parametric arc costs	$c_\lambda(i, j) = (c_{ij} - \lambda \cdot b_{ij})$, where $b_{ij} = 0, 1$	$w_\lambda(i, j) = (\lambda \cdot k_{ij} - e_{ij})$, where $k_{ij} = 0, 1, 2, 3$
Decision to be found	To solve the problem for all λ in $(-\infty\ +\infty)$ and to find all the assignments	To find a single value λ^* and a single assignment, for which $L(\lambda)=0$
Practical application	To solve the problem of computing the characteristic max-polynomial of a matrix in the max-algebra.	To solve the problem of maximizing the average profit for a fleet of vehicles.

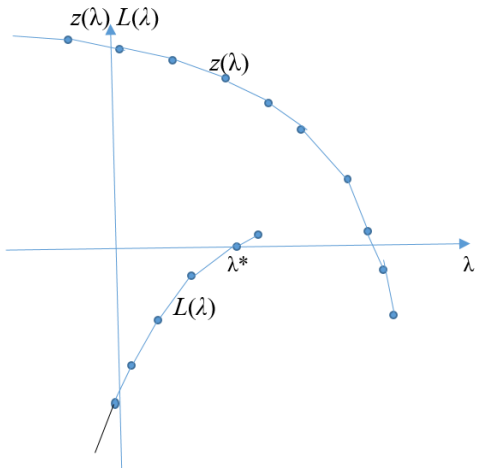

Figure 4. Graph of two objective functions, $z(\lambda)$ of the Gassner-Klinz problem and $L(\lambda)$ of the average fleet profit maximization problem

8.2 A brief review of the Gassner-Klinz algorithm and its adaptation

The GKA algorithm aims to solve the parametric assignment problem described in Table 8 for *all possible* values of parameter λ; its worst-case complexity is $O(n^3)$. that is, the same as that for the standard (non-parametric) linear assignment problem. In contrast to the Newton-type algorithm **A1,** the GKA does not solves each of $O(n)$ assignment problems appearing in critical points, but, rather, solves an assignment problem only once and then transforms the obtained assignment into a new one with certain local operations so that that the minimum-cost assignments for all the values of λ are found in $O(nm + n^2 \log n)$ time, where m and n are the number of arcs and the number of nodes, respectively, in an underlying bipartite graph described below. We will follow the same idea of the GKA and adapt it, adding necessary changes, in order to take into account that the objective functions in the two problems are different (see Table 8).

Since the two optimization problems in Table 8 are very close, it suffices to replace $c_\lambda(i, j) = (c_{ij} - \lambda \cdot b_{ij})$ in the Gassner-Klinz problem by $c'_\lambda(i, j) = (c'_{ij} - \lambda \cdot b'_{ij}) = (- e_{ij} - \lambda \cdot (-k_{ij}))$ and then use the existing algorithm GKA. In this case the slope of the parametric objective function in the derived problem (which, obviously, in this case will be our PAP) will be positive and will decrease with the growth of parameter λ until $L(\lambda)$-becomes zero, whereas the slope of the objective function in the original Gassner-Klinz problem is negative and its absolute value increases when the parameter increases to infinity (see Fig. 4). Another observation is that in the original Gassner-Klinz problem $b_{ij} \in \{0, 1\}$ whereas b'_{ij} are to be {0, -1, -2, -3}; however, this difference can be easily overcome and it does not influence the algorithm complexity. Finally, the starting point and the stopping rule are different in the GKA and in the modified algorithm. Indeed, when solving the PAP problem, one does not need to solve the assignment problems in all the critical points for all possible $\lambda \in (-\infty, +\infty)$; rather, the modified algorithm starts with a known lower bound of the parameter and stop as soon as the cost $L(\lambda)$ of a current assignment becomes zero.

Below, for the reader's convenience, we review the proposed modification of the GKA and, in order to give a more complete picture, describe the steps of the adapted algorithm. For easier comparison, in this subsection we borrow graph-theoretic terminology from [31] to describe the modified algorithm.

Consider the parametric assignment problem as a minimum-cost matching problem in a bipartite graph $G = (U, V, W)$, where the vertices U and V correspond, respectively, to the rows and columns of the matrix $\boldsymbol{W}(\lambda)$, $|U| = |V| = n$, $|W| = m$. Arc $(i, j) \in W$ only if arc (i, j) exists in the graph G_{gen}. Arcs lead from set U to set V. The parametric cost of arcs $(i, j) \in W$ is $w(i, j) = (\lambda \cdot k_{ij}$

- e_{ij}), where $i \in U$ and $j \in V$, i.e., the element of the matrix $w_{ij} \in \boldsymbol{W}(\lambda)$ is equal to the cost of the arc (i, j), $w(i, j) = w_{ij}$.

Let $A^*(\lambda')$ be a minimum-cost assignment in graph G for $\lambda = \lambda'$. Exactly as in [31], we associate with the graph G and assignment $A = A^*(\lambda')$ a residual graph $N(A) = (U, V, W^*)$ constructed as follows: All the arcs $(i, j) \in A$, where $i \in U$ and $j \in V$ are replaced by *backward arcs* (j, i) of cost $w(j, i) = - w(i, j) = (- \lambda \cdot k_{ij} + e_{ij})$.

Note that it is at this point that we make changes needed to be taken because coefficients k_{ij} in our objective function have the opposite sign compared with the corresponding b_{ij} in the GKA.

The remaining arcs, called *forward arcs*, as well as their costs, remain the same as in the graph G. Graph $N(A)$ has no cycles with a negative cost for $\lambda = \lambda'$. Let us increase parameter λ and let λ_c be the minimum value of λ such that in the residual graph $N(A)$ there is a cycle C with zero cost, $w(C, \lambda_c) = 0$, and $w(C, \lambda) < 0$, when $\lambda > \lambda_c$. Value λ_c and cycle C are the *critical point* and *critical cycle*, respectively. The assignment A remains a minimum-cost assignment in the interval $\lambda \in [\lambda', \lambda_c]$. Then, in the interval $\lambda \in [\lambda_c, \lambda'_c]$, where λ'_c is a critical point next to the critical point λ_c, the minimum-cost assignment changes to $A' = (A \setminus C') \cup (C' \setminus A)$, where C' is a subset of the arcs obtained from the C by replacing all the backward arcs with the corresponding forward arcs; the cost of A' changes to $w(A', \lambda) = w(A, \lambda) + w(C, \lambda)$. It is worth to notice at this stage that the following essential property is valid: when the parameter λ increases, the objective function of our problem also increases while the objective function in the Gassner-Klinz problem decreases (see Fig. 4). At point $\lambda = \lambda_c$ the costs of assignments A and A' are equal, $w(A, \lambda_c) = w(A', \lambda_c)$. By changing the cost and the direction of all the arcs in cycle C to opposite ones in the residual graph $N(A)$, one obtains the residual graph $N(A')$.

The following question is crucial: how, starting with the minimum-weight assignment A for $\lambda = \lambda'$, one can determine the critical point λ_c. Recall that in the residual graph $N(A)$, for $\lambda = \lambda'$ there are no cycles with a negative cost, and that, when the parameter λ increases, some cycle C with a zero cost appears only at the point $\lambda = \lambda_c$, $w(C, \lambda_c) = 0$. To determine this cycle, we follow the approach proposed in [31] and also use the parametric shortest path algorithm of Karp and Orlin [32] and its improved version in [33]. The steps of the modified GKA adapted for solving our **PAP** problem are the following:

Algorithm A2

Step 1. *Initialization*

1.1. Solve the initial standard assignment problem for the known matrix $\boldsymbol{K}$. Denote by I the obtained optimal (minimum-cost) assignment.
1.2 Calculate the average profit λ_0 achieved with the obtained assignment I:

$$\lambda_0 = \sum_{(i,j)\in I} e_{ij} / \sum_{(i,j)\in I} k_{ij}.$$

//This step is different compared with the corresponding initialization step in the original GKA.
1.3. Find the minimum-cost assignment $A = A^*(\lambda_0)$ in the graph G for $\lambda = \lambda_0$.
1.4. Construct the residual graph $N(A)$.

Step 2. *Finding the nearest critical point.*
//In this step we take into account that in the objective function the term $b'_{ij} = - k_{ij}$ is of opposite sign compared to the Gassner-Klinz assignment problem.

2.1. Apply the parametric shortest path algorithm to find the nearest critical point λ_c and the critical cycle C for which $w(C, \lambda_c) = 0$.
2.2. Calculate the cost of the assignment A at the point $\lambda = \lambda_c$:

$$w(A, \lambda_c) = \lambda_c \sum_{(i,j)\in A} k_{ij} - \sum_{(i,j)\in A} e_{ij}.$$

// When we increase the values of parameter λ from one critical point to the next one, we stop and go to Step 3.1 at the moment when we find (for the first time) a current assignment A such that $w(A, \lambda_c) \geq 0$. According to Proposition 4 in Section 4, this assignment A maximizes the average fleet profit per vehicle.

2.3. If $w(A, \lambda_c) \geq 0$, go to Step 3.
2.4. Create the new minimum-cost assignment $A' = (A \setminus C') \cup (C' \setminus A)$.
// C' is a subset of arcs obtained from C by replacing all the backward arcs with the corresponding forward arcs.
2.5. Convert graph $N(A)$ to graph $N(A')$; set $N(A) := N(A')$ and $A := A'$.
2.6. Return to Step 2.1.

Step 3. *Solution of the vehicle scheduling problem*
//This step is absent in the GKA because the max-benefit UAV fleet problem was not a subject studied by Gassner and Klinz.
//Let us denote by A^* the optimal assignment, which maximizes the average fleet profit per vehicle; $\sum_{(i,j)\in A^*} k_{ij}$ expresses the corresponding optimal number of the required vehicles.

3.1. Set the assignment that maximizes the average fleet profit per vehicle $A^* := A$.
Calculate the maximal average profit $P^* = \lambda^* = \sum_{(i,j)\in A^*} e_{ij} / \sum_{(i,j)\in A^*} k_{ij}$.
3.2. Calculate the optimal number of vehicles, which is $\sum_{(i,j)\in A^*} k_{ij}$, to meet the given vehicle schedule.
End.

Evidently, the adapted version of GKA has the same complexity as the original GKA, and, hence, the original max-benefit UAV fleet problem is solved in $O(n^3)$ time.

9. Conclusion

In this article we consider the problem of scheduling/assigning periodically repeated flights performed by a fleet of UAVs/drones. We extend the known scheduling/assignment problem for minimizing the number of aircraft to a more general problem of maximizing the average drone fleet profit per vehicle.

The main contribution of the present study is two-fold. First, we formulate and optimally solve a new bi-matrix average fleet profit maximization model, using profit and capacity matrices, which is a special case of airline fleet assignment problems widely used in the airline industry. Secondly, the aircraft fleet profit optimization problem is reduced to a special type of the parametric assignment problem; moreover, we find a way to speed up the solution time of a Newton type algorithm and provide an improved algorithm that solves the initial aircraft assignment problem in question in $O(n^3)$ time. In addition to the above, the special case problem solved in this study clearly not only has its own merits, but can also serve as a "building block" for solving more complex problems. For example, it can be used to solve the mathematical programming-based model for periodic airline fleet assignment proposed in [34] using branch-and-bound strategies. Indeed, the maximum average profit decision represents a "warm start" and can also be used as a comprehensive upper bound on the maximum expected total fleet profit less penalties and costs of fleet use, which is the common objective function in [34].

A challenging open question for further research is to find other solvable cases of the general max–profit fleet assignment/scheduling problem. We believe that the proposed graph approach to problem analysis and algorithm design can be extended and applied to efficiently solve other combinatorial periodic assignment/scheduling problems, such as minimizing fuel consumption and CO2 emissions, planning recovery operations for unexpected disruptions, scheduling periodic and sporadic tasks in real time and others.

Authors and affiliations

Vladimir Kats
Institute of Mathematics and Applications, Ben Gurion University, Beer-Sheva, Israel
vkats782@gmail.com; katzlu@bgu.ac.il

Eugene Levner (corresponding author)
Holon Institute of Technology, Holon, Israel
levner@hit.ac.il

Conflicts of Interest: The authors declare no conflicts of interest.

Author Contributions: Conceptualization, Vladimir Kats and Eugene Levner; Formal analysis, Eugene Levner; Investigation, Vladimir Kats and Eugene Levner; Methodology, Eugene Levner; Validation, Vladimir Kats and Eugene Levner; Visualization, Vladimir Kats; Writing – original draft, Vladimir Kats and Eugene Levner; Writing – review & editing, Eugene Levner.

Funding.This research received no funding.

References

1. Engesser, V., Rombaut, E., Vanhaverbeke, L. and Lebeau, P., 2023. Autonomous delivery solutions for last-mile logistics operations: a literature review and research agenda. *Sustainability*, *15*(3), p.2774.
2. Rejeb, A., Rejeb, K., Simske, S.J. and Treiblmaier, H., 2023. Drones for supply chain management and logistics: a review and research agenda. *International Journal of Logistics Research and Applications*, *26*(6), pp.708-731.
3. Dolgui, A., and Ivanov, D. (2021a). Ripple effect and supply chain disruption management: New trends and research directions. *International Journal of Production Research*, 59(1), 102–109. https://doi.org/10.1080/00207543.2021.1840148
4. Dolgui, A., and Ivanov, D. (2021b). 5G in digital supply chain and operations management: Fostering flexibility, end-to-end connectivity and real-time visibility through internet-of-everything. *International Journal of Production Research*, 1–10. https://doi.org/10.1080/00207543.2021.2002969
5. Shakhatreh, H., Sawalmeh, A.H., Al-Fuqaha, A., Dou, Z., Almaita, E., Khalil, I., Othman, N.S., Khreishah, A. and Guizani, M., 2019. Unmanned aerial vehicles (UAVs): A survey on civil applications and key research challenges. *IEEE Access*, *7*, pp.48572-48634.
6. Manfreda, S., McCabe, M.F., Miller, P.E., Lucas, R., Pajuelo Madrigal, V., Mallinis, G., Ben Dor, E., Helman, D., Estes, L., Ciraolo, G. and Müllerová, J., 2018. On the use of unmanned aerial systems for environmental monitoring. *Remote sensing*, *10*(4), p.641.
7. Maffezzoli, F., Ardolino, M., Bacchetti, A., Perona, M. and Renga, F., 2022. Agriculture 4.0: A systematic literature review on the paradigm, technologies and benefits. *Futures*, *142*, p.102998.
8. Garey, M., Graham, R., Johnson, D. (1976). Some NP-complete geometric problems. *Proceedings of the 8th Annual ACM Symposium on Theory of Computing*, 10-22.
9. Psaraftis, H.N., Wen, M. and Kontovas, C.A., 2016. Dynamic vehicle routing problems: Three decades and counting. *Networks*, *67*(1), pp.3-31.
10. Rave, A., Fontaine, P. and Kuhn, H., 2023. Drone location and vehicle fleet planning with trucks and aerial drones. *European Journal of Operational Research*, *308*(1), pp.113-130.

11. Zhang, K., Lu, L., Lei, C., Zhu, H. and Ouyang, Y., 2018. Dynamic operations and pricing of electric unmanned aerial vehicle systems and power networks. *Transportation Research Part C: Emerging Technologies*, *92*, pp.472-485.
12. Balac, M., Vetrella, A.R., Rothfeld, R. and Schmid, B., 2019. Demand estimation for aerial vehicles in urban settings. *IEEE Intelligent Transportation Systems Magazine*, *11*(3), pp.105-116.
13. Otto, A., Agatz, N., Campbell, J., Golden, B. and Pesch, E., 2018. Optimization approaches for civil applications of unmanned aerial vehicles (UAVs) or aerial drones: A survey. *Networks*, *72*(4), pp.411-458.
14. Attenni, G., Arrigoni, V., Bartolini, N. and Maselli, G., 2023. Drone-Based Delivery Systems: A Survey on Route Planning. *IEEE Access*, *11*, pp.123476-123504.
15. She, R.F., 2023. *Design of freight logistics services in the era of autonomous transportation* (Doctoral dissertation, University of Illinois at Urbana-Champaign).
16. Dantzig, G.B. and Fulkerson, D.R. (1954). Minimizing the number of tankers to meet a fixed schedule. *Naval Res. Logist. Quart.* 1, 217-222.
17. Ford, L.R. and Fulkerson, D.R. (1962). *Flows in Networks*. Princeton University Press, Princeton.
18. Karzanov, A.V. and Livshits, E.M. (1978). Minimal quantity of operators for serving a homogeneous linear technological process. *Automation and Remote Control,* 39, part 2, 538-542.
19. Kats, V.B. (1982). An exact optimal cyclic scheduling algorithm for multi-operator service of a production line. *Automation and Remote Control*, 43, part 2, 538-542.
20. Orlin, J.B. (1982). Minimizing the number of vehicles to meet a fixed periodic schedule. *Operations Research,* 30(4), 760 –776.
21. Kats, V., and Levner, E. (1997). Minimizing the number of robots to meet a given cyclic schedule. *Annals of Operations Research*, 69, 209-226.
22. Kats, V. and Levner, E. (1998). Minimizing the number of vehicles in periodic scheduling: The non-Euclidean case. *European Journal of Operational Research,* 107(2), 371-377.
23. Orlin, J.B. (1984). Minimum convex cost dynamic network flows. *Mathematics of Operations Research*, 9(2), 190-207.
24. Campbell, A.M. and Hardin, J.R. (2005). Vehicle minimization for periodic deliveries. *European Journal of Operational Research*, 165(3), 668-684.
25. Kochenberger, G.A., Glover, F., Alidaee, B. and Rego, C. (2004). A unified modeling and solution framework for combinatorial optimization problems. *OR Spectrum*, 26(2), 237-250.
26. Kabadi, S.N. and Punnen, A.P. (2008). A strongly polynomial simplex method for the linear fractional assignment problem. *Operations Research Letters*, 36(4), 402-407.
27. Megiddo, N. (1979). Combinatorial optimization with rational objective functions. *Mathematics of Operations Researh*, 4, 414–424
28. Gallo, G., Grigoriadis, M. D., and Tarjan, R. E. (1989). A fast parametric maximum flow algorithm and applications. *SIAM Journal on Computing*, *18*(1), 30-55.
29. Shigeno, M., Saruwatari, Y., & Matsui, T. (1995). An algorithm for fractional assignment problems. *Discrete Applied Mathematics*, 56(2-3), 333-343.
30. Radzik, T. (1992). Newton's method for fractional combinatorial optimization. In *Proceedings of the 33rd Annual Symposium on Foundations of Computer Science*, 659-669.
31. Gassner, E. and Klinz, B. (2010). A fast parametric assignment algorithm with applications in max-algebra. *Networks: An International Journal*, 55(2), 61-77.
32. Karp, R. M. and Orlin, J. B. (1981). Parametric shortest path algorithms with an application to cyclic staffing. *Discrete Applied Mathematics,* 3(1), 37-45.

33. Young, N.E., Tarjan, R.E., & Orlin, J. B. (1991). Faster parametric shortest path and minimum balance algorithms, *Networks*, 21, 205–221.
34. Belanger, N., Desaulniers, G., Soumis, F. and Desrosiers, J. (2006). Periodic airline fleet assignment with time windows, spacing constraints, and time dependent revenues. *European Journal of Operational Research*, *175*(3), 1754-1766.