Minimizing User Inconvenience and Operational Costs in a Dial-a-Flight Problem for Air Safaris

Jean-François Cordeau[a], Manuel Iori[b], Nilson F.M. Mendes[b], Davide Nelli[b], Riccardo Tedeschi[b]

[a] Department of Logistics and Operations Management, HEC Montréal, Canada;
[b] Department of Sciences and Methods for Engineering, University of Modena and Reggio Emilia, Italy

ARTICLE HISTORY
Compiled December 8, 2021

ABSTRACT
We study a Dial-a-Flight Problem faced by one of the major safari airline companies in Tanzania. Given a set of daily passenger requests and a fleet of heterogeneous airplanes, the problem requires to determine the best set of itineraries to transport the passengers from their origins to the requested destinations within specific time windows, while satisfying a number of operational constraints. The aim is to minimize user inconvenience, measured by delays with respect to the pre-defined time windows and by the number of intermediate stops in the itineraries, and operational cost. The problem is complicated by the high number of daily requests in peak touristic periods, and by the fact that refueling is possible only at a limited number of airstrips. We solve the problem by means of an adaptive large neighborhood search, which we enrich with local search operators and a set partitioning model. Extensive computational tests on real-world instances prove the effectiveness of the proposed algorithm, which can improve the solutions found by the company both in terms of operational cost and user inconvenience, in reasonable computational time.

KEYWORDS
Dial-a-Flight Problem; Safari; Adaptive Large Neighborhood Search; Set Partitioning

1. Introduction

Tanzania is an Eastern African country with many tourist attractions, as national parks, conservation areas, reserves and marine parks. These include popular tourist destinations, such as the island of Zanzibar and the UNESCO World Heritage site of Mount Kilimanjaro. These attractions make tourism the largest foreign source of income for the country, contributing with an average of 2 billion U.S. dollars per year since 2012, which is roughly equivalent to 25% of all foreign exchange earnings. Tourism also contributes to more than 17% of the national gross domestic product, creating more than 1.5 million jobs and being the fastest growing sector (National Bureau of Statistics, Tanzania 2020).

Many pioneering entrepreneurs were quick in identifying tourism as the true national vocation of Tanzania. In early years, tourism gave life to a great number of vehicle-

CONTACT A. Manuel Iori. Email: manuel.iori@unimore.it

based safari companies, which opened the country to a new surge of visitors. It is around 30 years ago that some companies recognized the opportunity to develop an airline safari network capable of accessing the most remote parts of the country, offering more agile and comfortable services to the visitors.

Currently, these companies operate on a network composed by more than 100 airstrips, connecting not only the main cities and tourist sites in the country, but also far-away destinations located in the middle of the park areas (see Figure 1 for a condensed view of Tanzania's airstrip network). Flights between these airstrips are performed by means of fleets of small airplanes, each transporting around a dozen passengers. For most companies, flights are organized on a daily basis, combining in the best possible way the travel bookings received. The transport is done under tight constraints imposed by hard operational conditions, including the lack of refueling options in many of the airstrips, and the need to provide a high level service to the customers.

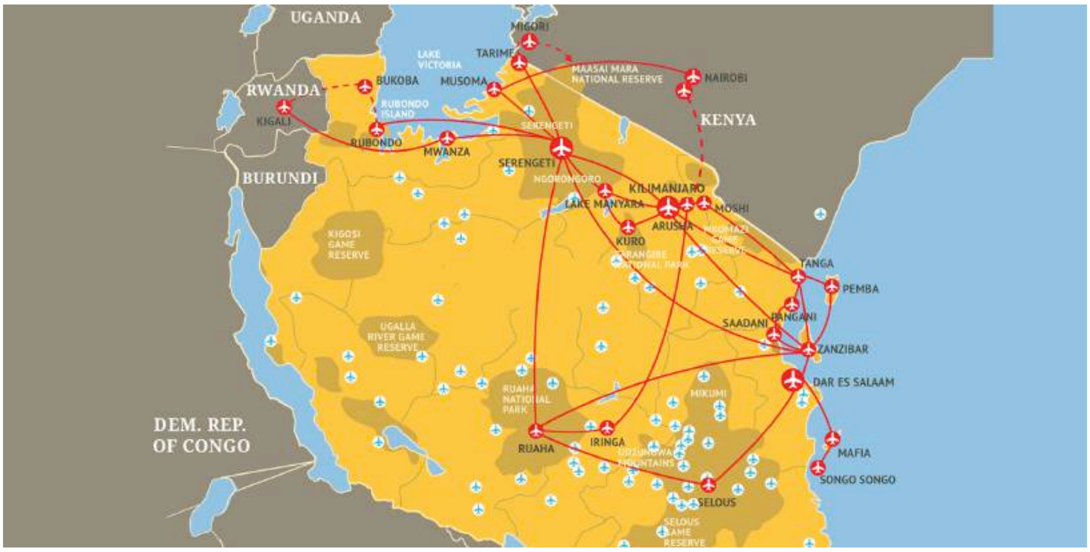


Figure 1. Main airstrip network in Tanzania (image taken from http://www.coastal.co.tz/)

This article studies the operations of one of the major safari airline companies in Tanzania. The company has a fleet of around 20 airplanes comprising two models, the high-wing braced cabin monoplane Cessna Caravan-208B and the single-engine turboprop Pilatus PC12. The two models have a similar capacity in terms of passengers transported, but significant differences in speed, fuel consumption, cabin comfort, and maximum cargo weight. Each airplane starts from a given airstrip and may end its daily sequence of flights at a different airstrip.

Transport requests for a certain day are collected until the previous day and then processed, at the company headquarters, so as to form the best set of flights. Each request consists of an itinerary from a given origin airstrip to a given destination airstrip, to be performed within a specific time window both at the pickup and at the drop-off sites. Possible violations of the time windows are accepted but penalized (in other words, these are *soft* time windows). All clients should be able to reach their destination with a maximum of three intermediate stops. The number of intermediate stops is not only a constraint, but also a crucial parameter to be taken into account in the evaluation of the user inconvenience. Passengers may also select two different, and not exclusive, additional classes of service: a passenger in *fast class* is guaranteed to reach his destination without any intermediate stop; a passenger in *extra-luggage class* has the right to transport an additional piece of luggage on board, resulting in a maximum of 30 kg instead of 15 kg. At most airstrips, takeoffs and landings must be performed in daylight as there is no artificial lighting system. This corresponds to a *hard* time window that cannot be violated. Fuel is available only at certain airstrips. In addition, for safety reasons, a minimum quantity of fuel after landing and a maximum weight before takeoff are imposed.

The aim of the company is to combine the daily requests into a set of flights, in such a way that: (i) all requests are fulfilled; (ii) all operative constraints are satisfied; (iii) user inconvenience, measured by delays in the time windows and number of intermediate stops, is minimized; and (iv) operational costs as well are minimized. Operational costs (which are discussed in detail in Section 2 below) are caused by daily fees for the use of an airplane, number of kilometers traveled, refueling, and landing fees at airstrips. In the following, we refer to this problem as the *Dial-a-Flight Problem for Air Safaris* (DAFPAS).

The DAFPAS belongs to the category of *Dial-A-Flight Problems* (DAFP), a class of problems that has received increasing attention in recent years. The DAFP is indeed one of the key problems arising in air passenger transportation (Espinoza et al. 2008a,b), and differs from other classical airline scheduling problems (see, e.g., Klabjan 2005) because the planning changes on a daily basis instead of making use of structured medium-term or long-term schedules. The DAFP is more similar to the well-known *Dial-a-Ride Problem* (DARP), which is usually studied in the context of ground transportation vehicles (Cordeau and Laporte 2007; Doerner and Salazar-González 2014), and to other transportation on demand problems (Cordeau et al. 2007), with which it shares the need of identifying a user inconvenience function. All such problems are not only NP-hard, but also very difficult in practice, and instances of large size cannot be solved exactly within limited computing times.

The aim of this paper is to propose a methodology to quickly obtain good-quality solutions for the real-world DAFP that we describe. To this aim, we develop an *Adaptive Large Neighborhood Search* (ALNS) metaheuristic, which relies on several destroy and repair mechanisms. We also embed into the ALNS a set of local search procedures to explore more intensively the neighborhoods around promising solutions, and adopt a *Set Partitioning* (SP) model to iteratively post-optimize the pool of routes that has been built during the search. Recently, ALNS methods have obtained very good results on a large number of vehicle routing problems (see, e.g., Ropke and Pisinger 2006 and Pisinger and Ropke 2010). Still in the field of routing, the combination of ALNS with local search have led to good results (as in, e.g., Dell'Amico et al. 2016), as well as the use of SP models as post-optimization tools (as in Subramanian et al. 2013). Convincing results have also been obtained by similar approaches on the closely related DARP, by local search based metaheuristics (Parragh et al. 2010; Masmoudi et al. 2017) and by ALNS algorithms (Masson et al. 2013; Gschwind and Drexl 2019). In our study, the combined use of ALNS with local search and an SP model led to prominent computational results, achieving good-quality solutions with limited computational effort.

The main contributions of this paper are as follows:

- We describe in detail a real-world transportation on demand problem and we contrast it with the existing literature. The interest derives not only from the particular application at hand, but also from the fact that the problem is very general and may well represent several other situations arising in passenger transportation.

- We perform a computational study of the user inconvenience function, with a focus on the number of intermediate stops. An economic interpretation of the user inconvenience has been defined in agreement with the company.
- We design a new metaheuristic based on the ALNS paradigm. Some operators have been adapted from the existing literature, whereas others (inter-move and parallel set partitioning) have been newly designed on the basis of the specific characteristics of the problem at hand.
- We use an SP model to attempt recombining the routes explored during the ALNS search. The model is used in an iterative manner, with the aim of determining the best balance between ALNS and SP computational efforts.
- We present extensive computational tests on a set of real-world instances. The outcome shows that the presented algorithm is effective in improving the solutions found by the company, achieving lower cost and lower user inconvenience, on average, within limited computational times, and hence can be considered as a good solution tool for the problem.

The remainder of this paper is organized as follows. Section 2 provides a detailed description of the DAFPAS. Section 3 reviews the related scientific literature. A formal mathematical formulation of the problem is provided in Section 4. Section 5 contains the details of the metaheuristic algorithm that we developed. The result and analysis of extensive computational experiments that we performed on a set of real-world instances are given in Section 6. Some final conclusions and future research avenues are drawn in Section 7.

2. Problem Description

The DAFPAS lies in the class of DAFP, but contains a number of specific characteristics induced by its application context. In this section, we describe the problem characteristics in detail and present some assumptions that we made to produce a good model.

2.1. Airstrips and Network

We are given a set of 21 airstrips, that represent the vertices of a complete undirected graph. Each airstrip is characterized by the fact that it can be used for refueling or not. Each airplane landing at an airstrip without refueling should always have in its tank a minimum quantity of fuel, imposed by security rules. The path to be followed for flying from an airstrip to another is known, and so is the distance to be traveled and the expected flight time and fuel consumption.

For each airstrip, a maximum allowed weight at take off for each airplane (see also Section 2.3) and a landing fee are imposed. A minimum time in which an airplane is required to remain on the ground between a landing and the next takeoff is also imposed. This time, called *ground time* (and being around 20–30 minutes in our real-world application), depends on the airstrip and comprises the operational time for alighting/boarding passengers, the possible need for refueling, and a break time for the pilot. A relevant characteristic of our case study is that the ground time is always large enough to allow the refueling of an airplane, when needed.

Operations at an airstrip are allowed only within a daily time window. Indeed, for most of the airstrips, takeoffs and landings must happen in daylight. We thus set the operating time window to [6:00 am, 6:30 pm] in our tests. For some other airstrips, namely, Dar es Salaam and Arusha (see Figure 1), earlier departures starting from 5:30 are allowed because of the presence of artificial light and an air control tower.

2.2. Requests

We are given a set of a transportation requests, each with given origin and destination airstrips. The majority of requests is associated with a single passenger. Under this assumption, it is possible that passengers that booked the air safary as a group be split into different flights. This is allowed by our methodology, and by the company as well, but it rarely occurs in the solutions because optimization tends to group passengers with the same origin, destination and time windows on the same flights. Furthermore, the presence of large groups is very limited in the instances that we solved, in which the majority of requests is composed by pairs of customers. Some requests (around 14% of the total) correspond to families with children, who cannot be separated from a parent. In such cases, group splitting is forbidden. Each passenger is characterized by a type (male, female, or child) and a class (standard, fast-class, extra-luggage, or combined fast-class and extra-luggage). This information is used to compute the expected weight of a request and the maximum number of intermediate stops between pickup and drop-off.

Each request has a specific time window on the arrival times both at the pickup and at the drop-off airstrips. This represents a deviation from standard DARPs where a time window is imposed on either the pickup or the delivery site, in combination with a maximum ride time. The reason for this deviation is that target arrival times at pickup and drop-off locations are already defined at the time of booking. These target times are used by the customers (or by their tour operator) to coordinate the ground activities of the safari, which are not operated by the air company. The time windows are simply intervals set around these target arrival times. In our case study, the intervals last one hour (half an hour before and half an hour after the target time) and correspond to a discrepancy with respect to the target that is supposed to be accepted without inconvenience by the clients. Early or late arrivals with respect to these time windows are still accepted, but penalized (see Section 2.5 for details). In such cases, passengers are required to be at the pickup locations at the new arranged pickup time, and ground transportation services that will take care of the passengers after their landing should be at the drop-off locations at the new arranged arrival time. Both times are communicated the day before. Note that it is not expected that an airplane wait in case of early arrival. We simply suppose that passengers and ground services will be able to reach the new airstrip within the new arranged time, as this was communicated in advance. Waiting is allowed in case it helps minimize the total user inconvenience of a route (and this is indeed at the basis of one or our solution methods in Section 5 below). Considering the size of the instances we tackle, we observe that the number of requests per day may vary from about 100 requests in the months of lowest demand, to about 350 requests in the months of highest demand.

2.3. Airplanes

We are given a heterogeneous fleet of airplanes. Each airplane has a certain cruise speed, which determines the traveling time between two airstrips, and a certain fuel consumption, discussed in detail in Section 2.4. An airplane is also characterized by a number of seats for passengers, a maximum allowed total weight for taking off, and a maximum fuel capacity. In real applications, the total weight for taking off depends not only on the airplane, but also on the airstrip, because airstrips with higher altitude offer lower air lift. As the differences between the airstrips are very small in our case study, we assumed that each airplane has a unique value of maximum weight for taking off.

Note that the weight of an airplane depends on both the transported passengers and the quantity of loaded fuel, and this represents a key decision variable when building routes. This happens in any air transportation problem (Cordeau et al. 2007), but is particularly relevant in our study because refueling is not available in most of the airstrips and the airplanes have a small size. We can decide to load more passengers at the expense of a lower fuel tank level, or, vice versa, to cover longer distances at the expense of a reduced number of transported passengers. Note also that, after landing at an airstrip, an airplane must have at least the minimum quantity of fuel required to reach the closest airstrip (in case landing, for any reason, cannot be performed at the planned airstrip). This is imposed on each intermediate stop and also on the last stop of a day, to avoid having an airplane stuck at an airstrip without sufficient fuel.

Each airplane is also associated with a home location. At the beginning of a day, the airplane is available at the airstrip at which it ended the last flight on the previous day. At the end of the day, it can be located at any airstrip (provided it has sufficient fuel). Each airplane should be back to its home location at least once every three days. This is imposed to perform a required periodic maintenance on the airplane. In our study, we relaxed this assumption, supposing that each day is independent from the others. However, going back to the home location can be obtained, in our solution method, by imposing this location as the last arrival of the day for an airplane. For considering, instead, the original constraint as in the real-world application, one should model the problem as a dynamic multi-period problem, to be solved with a rolling horizon approach. The dynamic component is caused by the fact that requests for day $t+1$ would be known only when the flights for day t are already being operated. Note also that this multi-period aspect is related to the working shifts of the pilots, which we do not explicitly take into consideration in our model. Daily working shifts are disregarded because in practice pilots can be changed at airstrips, as it commonly happens in ground public transportation problems, and because the maximum working shift for Tanzanian rules is quite large (11 hours, see Civil Aviation, Tanzania 2017). It might be an interesting aspect, however, in the study of multi-period planning. The resulting problems are left as future research directions (see Section 7).

Concerning the size of the problem, the fleet owned by the company is composed by around 20 airplanes. In our instances, the number of available airplanes varies between 10, in the months with lowest demand, to 15, in the months with highest demand. The remaining airplanes are either unused and parked at their home locations, or in maintenance, or rented for private safari flights. The use of private flights is quite common for groups aiming at a higher service level (at a higher cost). It is implicitly assumed that the number of airplanes is large enough to satisfy all requests, as capacity is taken into account when accepting the bookings. Should there be an excess of requests or a shortage of airplanes, other airplanes could be rented on the market.

2.4. Costs

As reported by Cordeau et al. (2007), most studies on transportation on demand problems fall under two categories: minimizing costs subject to full demand satisfaction and side constraints; or maximizing satisfied demand subject to vehicle availability and side constraints. Our problem belongs to the first category. The costs we incur for serving the requests are the cost of daily use of each airplane, the cost of the mileage traveled, the fuel consumption, and the cost for the landings. Let us explain them in more detail.

It is quite common for airline companies with seasonal demand to lease the airplanes for long periods. In case the number of transportation requests increases, new airplanes are rented. As far as we are concerned, we deal with a problem where the company has at its disposal the entire fleet of airplanes. In the short-term, this means that the fixed cost of each airplane is only composed by its flight tax. Every day, and for each used airplane, the company pays a mandatory fee to the *Tanzania Civil Aviation Authority* (TCAA). Clearly, any used airplane has a certain fixed cost whose value does not affect the cost of a route. Nevertheless, the fixed costs gain in importance in the long-term view, especially for determining the economical sustainability of the activity. For this reason, in our computational tests (see Section 6 below) we evaluated a scenario in which the daily cost simply amounts to its mandatory fee, and another scenario in which it also includes the daily operating cost composed of staff salary and amortization. For comparison purposes, we also evaluated a third intermediate scenario.

It is common for most of the companies operating routing services with fleets of vehicles to impose a cost for each kilometer traveled. This also happens for our case study, where this cost is considered independent from the type of airplane and takes into account the direct expenses and maintenance.

Refueling may take place at a limited set of airstrips. The cost of the gasoline changes from one location to another, although very slightly. In our model, we assumed it to be equal for all airstrips, so as to simplify the refueling evaluation. The fuel consumption cost is then determined as the cost per liter multiplied by the number of liters consumed. The determination of the fuel consumption for an airplane is a complex task that would require to consider flight conditions (e.g., weather) and the path between departure and arrival airstrips (e.g., acceleration, deceleration, turns, difference in altitudes). We decided to adopt the same simplified criterion adopted by the company for the evaluation of the fuel consumption. We simply use a given linear consumption for flights that last an hour or less. By multiplying the number of traveled minutes by this parameter, we obtain the total liters consumed in a flight. For longer flights, the consumption in the first hour is computed using the first parameter, and the remaining consumption is obtained by multiplying the remaining time by a second, smaller, fuel consumption parameter. The second parameter is smaller than the first, because in shorter flights fuel consumed during landing and takeoff has a higher impact on the overall fuel consumed and also because the airplane does not reach a high altitude, thus encountering a higher air resistance.

Any time an airplane lands, a corresponding landing fee should be paid to the TCAA. This fee is independent from the airstrip at which the landing occurred.

2.5. *User inconvenience*

User inconvenience is a measure of the dissatisfaction of a passenger, and should be minimized together with the operational costs. In our case, in accordance with the company, we opted to measure the number of intermediate stops, and the amount of violation of the time windows both at the pickup and at the drop-off locations. The intermediate stops are already limited to a maximum of three for a standard passenger and to zero for a fast-class one, but on top of that they should also be minimized, as highly disliked and also increase the operational costs, due to the airport landing taxes and higher fuel consumption on takeoffs. The time window violation is considered both for early arrivals (as the difference between the earliest time and the arrival time if the arrival occurs before the earliest time) and for late arrivals (as the difference between the arrival time and the latest time if the arrival occurs after the latest). The total time window violation, expressed in minutes, is multiplied by a first penalization parameter, and the total number of intermediate stops by a second parameter. The values of these parameters have been established on the basis of discussions with the company, but, in addition to that, we performed extensive tests to assess their impact on the solutions.

We note that the time window violation is related to total riding time, waiting time and duration, which are other measures adopted in the DARP context where a single time window is imposed on the pickup (Cordeau and Laporte 2003). The time window imposed on the delivery represents, in this sense, a relevant difference between DARP and DAFPAS. This is a crucial constraint in our problem, because most airstrips are not equipped with structures to accommodate passengers, as they are just flat terrains in the middle of the parks, so passengers cannot be left on their own there. We also note that the company allows, in some cases, transshipment of passengers from an airplane to another in order to reach the required destinations. This is even more disliked by passengers, for obvious reasons. To look for low user-inconvenience solutions and obtain a simpler model, we disregarded the possibility of transshipment in our methodology. Fortunately, we were able to satisfy all requests on all instances even without transshipment. In Section 6, in order to evaluate the cost of the company solutions, transshipments, if any, have been penalized doubly compared to the cost of an intermediate stop.

3. Literature Review

The DAFPAS lies in the class of *Pickup and Delivery Problems* (PDPs), where requests are characterized by a point in which they need to be collected and a second point where they have to be delivered (see Battarra et al. 2014 and Doerner and Salazar-González 2014 for recent surveys). Among PDPs, the closest problem to the DFAPAS is the DARP, which requires to meet pickup and delivery transport demands by using a fleet of ground vehicles while minimizing cost and user inconvenience. The number of papers devoted to the solution of practical DARP has risen consistently in recent years, as can be noticed in, e.g., Cordeau and Laporte (2007) and Ho et al. (2018). Important differences arise between the DARP and the DAFPAS, in the definition of the constraints (e.g., there is no fuel restriction for the DARP), of the costs (e.g., there is no landing fee for the DARP) and of the user inconvenience (which is more related to time spent on board for the DARP, and on number of intermediate stops for the DAFPAS).

We can include the DAFPAS in the areas of Transportation on Demand and Air

Transportation. Transportation on demand concerns the relocation of passengers or goods between given origins and destinations, following specific requests by the users. Cordeau et al. (2007) give a description of this area, providing mathematical models for DARP services, urban courier transportation, ambulance fleet management, as well as static and dynamic DAFP. Air transportation is a wide area of research characterized by a variety of optimization problems. We refer the interested reader to the surveys by Barnhart et al. (2003) and Lacasse-Guay et al. (2010), to the book by Wensveen (2016), and to the recent case studies by Cacchiani and Salazar-González (2020) and Parmentier and Meunier (2020).

A number of relevant works combine air transportation with transportation on demand. Desaulniers et al. (1997) solved the daily aircraft routing and scheduling problem, which consists in constructing daily schedules for a heterogeneous aircraft fleet, with the aim of minimizing the fixed cost for each aircraft, the cost of the fuel consumed and the salaries of the crew members. They proposed SP and time constrained network flow formulations, and obtained good results by employing column generation. Keskinocak and Tayur (1998) addressed the time-shared jet aircraft scheduling problem, which can be seen as a DAFP where each aircraft can serve only one customer at a time. They studied the problem complexity and proposed solution methods based on Dynamic Programming (DP) and Mixed Integer Linear Programming (MILP). Ronen (2000) developed a decision support system based on the use of an SP model for scheduling charter airplanes. They minimized an objective function that included a number of operational costs and penalties for violations of soft constraints.

Martin et al. (2003) considered on-demand aircraft schedules for the so-called fractional aircraft programs (FAP). In a FAP, fractional owners purchase portions of specific aircraft from a management company, based on the number of actual flight hours they need. They are guaranteed access to an aircraft whenever and wherever they need it, by booking their service in advance. The authors present a management system that includes a MILP model for scheduling the aircraft. Similar FAPs were later studied by Yao et al. (2008), who discuss strategic planning issues, such as aircraft maintenance, crew swapping, and methods to increase and differentiate demand, and by Yang et al. (2008), who propose a scheduling decision support tool based on exact and heuristic algorithms aimed at increasing aircraft utilization.

Fagerholt et al. (2009) consider an air taxi service in Norway. Air taxi is an on-demand service in which customers can book in advance seats on aircraft operating on small regional airports. They presented a strategic decision support tool that helps estimate the trade-off between fleet size and service by heuristically solving an underlying DAFP. A similar problem involving a Belgian company has been studied by Van der Zwan et al. (2011), who developed an SP model. Very recently, Munari and Alvarez (2019) considered a FAP in which the aim is to determine airplane routing and scheduling to fulfill a list of flight requests. They propose a compact MILP model that takes into account mandatory aircraft maintenance and possible flight upgrades.

A typical feature of the DAFPAS is the limited fuel capacity. Other optimization problems with this feature have been considered in the literature. That is the case, for instance, in the Green Vehicle Routing Problem, which concerns fleets composed by alternative fuel-powered vehicles and helps in overcoming difficulties due to limited vehicle driving range in conjunction with limited refueling infrastructure (see, e.g., Bektaş et al. 2016). A restricted operational range caused by limited fuel capacity is a major concern also in military applications. Solution approaches in this field have investigated the use of aerial refueling, as in, e.g., Yamani et al. (1990), Yuan and Mehrez (1995) and Kannon et al. (2015).

A closely related transportation problem concerns on demand routing of helicopters. This topic has received a good amount of attention in recent years, especially for what concerns the transportation of rig crews in oil and gas offshore platforms, which was studied, among others, by Fiala Timlin and Pulleyblank (1992), Menezes et al. (2010), Qian et al. (2012), Hermeto et al. (2014) and de Alvarenga Rosa et al. (2016). As in our problem, using alternative vehicles, like vessels, is not an option because of low speed combined with long distances that need to be covered. The number of stops is also considered, not because of user inconvenience but for security reasons, as takeoffs and landings are dangerous on offshore platforms.

To the best of our knowledge, the term DAFP originates from the works of Espinoza et al. (2008a,b). Important differences arise between their problem and the one we face, as they can refuel at any airport, and they control user inconvenience by imposing hard constraints on maximum transit time and allowing at most one intermediate stop. In Espinoza et al. (2008a), the problem is modeled with a multicommodity network flow, having a direct flight for each pair of airports (a, b) and each departure time at a, and indirect flights (with one intermediate stop) for each triplet of airports (a, b, c) and each pair of departure times at a and b. The size of the network grows quickly with the number of airports, so they use aggregation techniques. They solve to proven optimality instances with up to six airplanes. In Espinoza et al. (2008b), they consider larger instances. They develop a parallel local search heuristic that invokes the multicommodity model for smaller instances containing a limited number of airplanes. Their approach is not practically replicable to our case study because it is based on the strict assumption that at most one intermediate stop occurs for each passenger. In related work, Engineer et al. (2011) introduce a column generation approach making use of a DP that operates on the time-expanded network underlying the previous multicommodity flow model. They use arc-based resource relaxation, forward and backward search, and a quick completion heuristic. They provide solutions for instances with up to 200 airplanes. This approach too depends on the assumption that at most one intermediate stop is allowed.

4. Problem Notation and Mathematical Formulation

We are given a complete undirected graph $G = (V, E)$, where V is the set of airstrips and E the set of edges connecting all pairs of airstrips. Each airstrip $i \in V$ is associated with an operating time window $[\tilde{e}_i, \tilde{l}_i]$, with a minimum ground time γ_i, and with a binary parameter r_i taking value 1 if and only if it is possible to refuel in i. The minimum ground time is large enough to allow a complete refueling of the airplane. With each edge $(i, j) \in E$ we associate a distance d_{ij}, a cost c_{ij}, a fuel consumption g_{ij}, and a traveling time t_{ij}.

We are also given a set S of n requests. Each request $s \in S$ has a pickup airstrip $s^+ \in V$, a delivery airstrip $s^- \in V$, a number of passengers π_s, a weight w_s, a maximum number of intermediate stops σ_s, and tentative pickup and delivery time windows $[e_{s^+}, l_{s^+}]$ and $[e_{s^-}, l_{s^-}]$, respectively. As explained in Section 2.2, these two time-windows are both imposed on the arrival time of the airplane at the node. Requests are satisfied by a fleet F composed of m airplanes. Each airplane $f \in F$ is located at airstrip $f^+ \in V$ at the beginning of the day, and is associated with a maximum number of passengers Π_f, a maximum fuel capacity G_f, and a maximum weight capacity W_f. The weight capacity should not be exceeded by the weighted sum of both passengers and fuel.

Let Ω^f be the set of all routes of an airplane $f \in F$. For simplicity, let $r \in \Omega^f$ define both a route and the corresponding route index. A route is defined as a sequence of visited airstrips $r = (r_1, r_2, \ldots, r_{|r|})$ together with a subset $S(r) \subseteq S$ of requests that are serviced by r. Each route satisfies weight limits, passenger limits, and all other applicable constraints. The first airstrip r_1 corresponds to the starting depot f^+ of f. Let $V(r) = \{i \in V : i \in r\}$ denote the set of airstrips visited by r, and note that $|V(r)| \leq |r|$ because an airstrip might be visited multiple times by a route. Let also $E(r) = ((r_1, r_2), (r_2, r_3), \ldots, (r_{|r|-1}, r_{|r|}))$ be the sequence of edges traversed by the route. The landing fee at an airstrip is denoted by c_ℓ, and the daily flight fee for an airplane by c_φ. By defining the cost per kilometer as c_d and the cost per liter of fuel as c_g, the total cost of a route is consequently given by

$$c_r^f = c_\varphi + c_\ell(|r| - 1) + \sum\nolimits_{(i,j)\in E(r)} (c_d d_{ij} + c_g g_{ij}). \tag{1}$$

Let $r(s^+)$ denote the index of the vertex of r at which the pickup of s occurs, and $r(s^-)$ the index of the vertex at which the delivery occurs. Let $\psi_s = r(s^-) - r(s^+) - 1$ be the number of intermediates stops for s. Let $k \in \{1, 2, \ldots, |r|\}$ be the index of a stop in r. Let also $a(k)$ denote the time at which the airplane arrives at the airstrip associated with vertex $r_k \in r$, considering $a(1)$ as the time at which the airplane is ready for departing at depot r_1. If an airplane flies from the airstrip associated with stop k to the airstrip associated with stop $k+1$, then the arrival time at $k+1$ is $a(k+1) = a(k) + w(k) + t_{r_k, r_{k+1}}$, where $w(k)$ is the total ground time during which the airplane remains grounded at r_k. The value of $w(k)$ is computed as part of the solution by the heuristic algorithm described below in Section 5. The value of $w(k)$ is never lower than the minimum ground time γ_i. It might be even higher than γ_i in those cases in which waiting might lead to a better user inconvenience function.

Under this notation, $a(r(s^+))$ and $a(r(s^-))$ give the arrival times at, respectively, the pickup and delivery site of request s. We can thus compute $\tau_{s^+} = \max\{e_{s^+} - a(r(s^+)); 0\} + \max\{a(r(s^+)) - l_{s^+}; 0\}$ as the time window violation, if any, at the pickup point of s. The value of τ_{s^+} takes into account both earliness, in its first component, and lateness, in its second component. Note that by computing the violation with respect to the arrival time, we are penalizing the waiting time, if any, of the airplane at the pickup point. Similarly, let $\tau_{s^-} = \max\{e_{s^-} - a(r(s^-)); 0\} + \max\{a(r(s^-)) - l_{s^-}; 0\}$ define the time window violation, if any, at the delivery point, and $\tau_s = \tau_{s^+} + \tau_{s^-}$ be the overall violation. Let ρ_ψ and ρ_τ be, respectively, the penalization factors associated with intermediate stops and time window violations. The total user inconvenience of route r is measured as

$$u_r^f = \sum\nolimits_{s\in S(r)} (\rho_\psi \psi_s + \rho_\tau \tau_s). \tag{2}$$

We can thus define the overall objective value associated with route r as

$$z_r^f = c_r^f + u_r^f. \tag{3}$$

To mathematically formulate the problem, we can represent each route as a column of an SP model. For airplane $f \in F$, and route $r \in \Omega^f$, let δ_{sr}^f be a binary parameter equal to 1 if request s is served by r, and 0 otherwise. Let y_r^f be a binary variable taking the value 1 if route r of airplane f is used, and 0 otherwise. The DAFPAS can be stated as

$$
\begin{aligned}
(\text{SP}) \quad & \min \sum_{f \in F} \sum_{r \in \Omega^f} z_r^f y_r^f && && (4)\\
\text{s.t.} \quad & \sum_{f \in F} \sum_{r \in \Omega^f} \delta_{sr}^f y_r^f = 1 && s \in S && (5)\\
& \sum_{r \in \Omega^f} y_r^f \leq 1 && f \in F && (6)\\
& y_r^f \in \{0,1\} && f \in F, r \in \Omega^f. && (7)
\end{aligned}
$$

Objective function (4) requires to minimize the sum of the route costs, computed using (3). Constraints (5) force each request to be served. Constraints (6) state that each airplane is used at most once, and constraints (7) give the variable domain.

For our instances, we find it convenient to consider airplane types instead of airplanes. Airplanes being of the same model and being located at the same airstrip at the beginning of the day are said to be of the same type. In other words, all airplanes of the same type can be interchanged as they can perform the same routes at the same cost. Now, we can re-consider the fleet F, originally composed by m airplanes, as a fleet F' composed by t airplane types, each having m^f airplanes, in such a way that $\sum_{f \in F'} m_f = m$. We can thus reformulate model (4)–(7) by replacing F with F' and substituting (6) with

$$\sum_{r \in \Omega^f} y_r^f \leq m_f \qquad f \in F'. \tag{8}$$

Despite this reduction, model SP remains very difficult to solve in practice because it contains an exponential number of columns. It can be used, however, in two different ways: (i) by solving the continuous relaxation of SP we can compute the reduced cost of a column, i.e., a route, and thus estimate how much this route could contribute to a complete solution; and (ii) by replacing the complete sets Ω^f of routes by smaller sets and solving the model to integer optimality, we can obtain a heuristic solution. Both approaches are employed in our metaheuristic method, as outlined in the next section.

5. Solution Methodology

To solve the DAFPAS, we implemented a metaheuristic that is based on iterated executions of an ALNS that is enriched with local search operators and an SP model. The method, called *Iterated ALNS* in the following, is summarized in Algorithm 1. It starts by creating a solution x with a constructive heuristic and a set of local search operators. The routes of this solution, represented by routes(x) in the pseudocode, are used to initialize an overall pool P of routes, which is going to be used later by the SP model. Then, the algorithm performs $iter_{\max}$ calls to the inner ALNS procedure. The first $iter_{phase1}$ times, the ALNS is invoked with an acceptance criterion that favors diversification and a stopping criterion that allows to perform a large search. In the remaining iterations, the ALNS is invoked with a more strict stopping criterion and with an acceptance criterion that favors intensification. Details on the adopted criteria and parameter values are given in Section 5.6 below.

Algorithm 1 Iterated Adaptive Large Neighborhood Search

---

procedure ITERATED ALNS
  $x \leftarrow$ Constructive Heuristic
  $x \leftarrow$ Local Search(x)
  $P \leftarrow$ routes(x) ▷ Pool of routes
  for $iter := 1$ **to** $iter_{\max}$ **do**
    if ($iter \leq iter_{phase1}$) **then**
      $x \leftarrow$ ALNS(x, P, acceptance_criterion_1, stopping_criterion_1)
    else
      $x \leftarrow$ ALNS(x, P, acceptance_criterion_2, stopping_criterion_2)
    end-if
  end-do
  return (x)

The core part of the solution method is the ALNS, whose pseudocode is provided in Algorithm 2. At step 1, the iteration counter t is set to 0 and some weight parameters to be used in the main ALNS loop are initialized. At step 2, it sets the current solution x_{curr} as the incumbent received in input. The main loop is performed until the stopping criterion received in input is met. It considers the current solution x_{curr} and modifies it by: (i) selecting destroy and repair operators according to the weights; (ii) applying these operators to perturb x_{curr}; and (iii) using local search to improve it. The destroy operator uses a degree of destruction d, randomly selected in a given interval. Any time a new solution is obtained, pool P is possibly enlarged with the new routes in the solution. The new solution obtained, x_{new}, is compared to the current one according to the input acceptance criterion. If the decision is to accept it, then x_{curr} is set to x_{new}. In such a case, we also check if x_{new} improves the incumbent solution and possibly update it.

After the main loop has been completed, the pool P is used to populate an SP model, invoked at step 19. This corresponds to the model outlined in Section 4, but invoked with the limited number of routes contained in P instead of all possible routes, and with a limited computational time. Before solving SP to integer optimality, with the aim of reducing the size of P and consequently the computational time required to solve SP, we first solve, at step 18, the linear relaxation of the model. We use the solution found to compute the reduced costs of all columns in the pool. The Θ columns of highest reduced cost, whose probability of entering the optimal SP solution is very low, are removed from P. However, we keep in the pool the columns corresponding to x_{best}, thus guaranteeing that the set partitioning problem still always includes a feasible vector of routes. The solution (x_{new}) found by SP(P) is then returned, and is possibly used to update the incumbent solution at step 20. In case the SP model execution is interrupted because it reaches the time limit and no solution is obtained, then x_{new} is set to a null solution and it will not update the incumbent. In the remainder of this section, we describe the details of each algorithmic component.

5.1. Constructive Heuristic

In the hope of quickly obtaining a first starting solution, we developed a constructive heuristic that is based on the concept of *cheapest insertion* and that builds routes in a parallel fashion. It opens m routes, one per airplane f, considering its departure airstrip f^+. Then, it attempts to extend the routes by inserting requests from S, one at a time. The requests are sorted in random order and then selected according to

Algorithm 2 Adaptive Large Neighborhood Search with local search and SP model

procedure ALNS(x_{best}, P, acceptance_criterion, stopping_criterion)
    set $t \leftarrow 0$ and initialize weights w_{mt}
    $x_{curr} \leftarrow x_{best}$
    while (stopping_criterion not met) **do**
        select destroy and repair method using weights w^d_{mt} and w^r_{mt}
        generate a degree of destruction $d \in [d_{\min}, d_{\max}]$
        $x_{new} \leftarrow$ Repair(Destroy(x_{curr}, d))
        $P \leftarrow P \cup \text{routes}(x_{new})$
        $x_{new} \leftarrow$ Local Search(x_{new})
        $P \leftarrow P \cup \text{routes}(x_{new})$
        if (Accept(x_{curr}, x_{new}, acceptance_criterion)) **then**
            $x_{curr} \leftarrow x_{new}$
            if $(z(x_{curr}) < z(x_{best}))$ **then** $x_{best} \leftarrow x_{curr}$
        end-if
        $t \leftarrow t + 1$
        update weights w_{mt}
    end-do
    solve $L(SP(P))$ and remove from P the Θ columns with higher reduced cost
    $x_{new} \leftarrow$ SP(P)   ▷ Set Partitioning model
    if $(z(x_{new}) < z(x_{best}))$ **then** $x_{best} \leftarrow x_{new}$
    return (x_{best}, P)

it. Their insertion in the routes is attempted by considering only a restricted set of positions. Let us consider a generic request s to be inserted in a route r. We define four types of insertions:

(i) s in inserted as first request in r. This insertion is attempted only if r is still empty. In such a case, r is expanded by including s^+ and s^-, one after the other, directly after f^+. The insertion of s^+ is skipped if $s^+ = f^+$. The starting time is computed so as to be exactly on time for the pickup in s^+;

(ii) s in inserted as last request in r, so s^+ and s^- are inserted at the end of r. The insertion of s^+ is skipped if it is equal to the last airstrip visited by r. The departure time at the beginning of r is not changed after the insertion, so cost and user inconvenience can be computed quickly;

(iii) s is inserted only if both s^+ and s^- are already contained in r. In this way, the airplane operating r needs no detour to pick up and drop off the additional passenger(s), but the feasibility of all constraints must still be checked;

(iv) s is inserted only if s^+ is already in the route, and in this case s^- is inserted as last airstrip visited by r.

Once a request has been selected, all routes are scanned with respect to the four defined types of insertion, and the one being feasible, if any, and having cheapest insertion cost is selected. The procedure is iterated until all requests have been served or there is no more space for further insertions. It is worth noting that no polynomial-time heuristic can guarantee to find a feasible solution for the DAFPAS, because this is a difficult task (indeed, just loading weights w_s into the airplanes by respecting capacities W_f is as hard as the classical bin packing problem). For this reason, we include the heuristic in a loop that is iterated, each time creating a new random order of requests, until a feasible solution is obtained. In our tests, we managed to obtain a feasible starting solution for each instance with at most two iterations of the loop. We also attempted, in preliminary experiments, some non-random orderings of the requests, especially looking at the time windows, but we did not obtain favorable results. The preliminary experiments also showed that the initial solution has a limited impact, as the successive ALNS components manage to improve its quality. At the end of the loop, if a feasible solution using strictly less than $|F|$ routes is found, the remaining unused routes are removed and the associated airplanes are simply kept in their original locations.

5.2. Destroy Operators

We have implemented *Random-removal*, *Worst-removal*, *All-removal*, and *Service time and Distance oriented removal* operators, which are quite common in the ALNS literature (Pisinger and Ropke 2010). All operators receive in input a solution x composed by $|\text{routes}(x)|$ routes and a percentage value d representing the degree of destruction. The value of d is randomly selected, at each ALNS iteration, in the interval $[d_{min}, d_{max}]$. In Random-removal, Worst-removal, and All-removal, d represents the percentage of the number of routes affected by the destruction. For *Service time and distance oriented removal*, d is the percentage of requests relocated. The output of a destroy operator is a partial solution where some requests and/or some airstrips have been removed from the preexisting routes. All routes obtained after destruction are elaborated in order to preserve feasibility. The removed requests will be reinserted by means of the repair operators.

Random Removal. It randomly selects, with uniform distribution, a route r in x. Then, it randomly selects a vertex r_i in r and removes it. All requests that depart from or arrive at r_i are removed as well from r. The route is then processed, so as to recompute costs and user inconvenience. The process is repeated $\lceil |\text{routes}(x)| d/100 \rceil$ times.

Worst Removal. Similarly to the previous operator, Worst Removal randomly selects a route r. In this case, however, the vertex r_i to be removed at each iteration is chosen as the worst vertex in the route, i.e., as the vertex whose removal would lead to the largest decrease in the route value. The decrease is computed in an approximated but quick way, as the saving that could be obtained by: (1) reducing the distance traveled by connecting directly r_{i-1} to r_{i+1}; (2) removing user inconvenience penalties associated with requests landing at or departing from r_i. The removal process is repeated $\lceil |\text{routes}(x)| d/100 \rceil$ times.

All Removal. It aims at a larger diversification with respect to the two previous operators. It selects a route r and then removes from r a certain number p of vertices, where p is a random number generated, with uniform distribution, between 1 and $|\text{routes}(x)|$. The process is repeated $\lceil |\text{routes}(x)| d/100 \rceil$ times.

Service time and Distance oriented Removal. A number of destroy operators in the literature, starting from Shaw (1997), attempt to remove requests that are close to one another, either in terms of distance, or time window, or both. The rationale behind that is to facilitate, later on, the work of the repair method. To this end, we define the *relatedness* of two requests, s and q, as $\delta(s, q)$, and compute it as the sum of the travel distances d_{s^+,q^+} and d_{s^-,q^-}, and of the time distances between the target pickup and delivery times of s and q. We start by selecting a route r at random. Then, we select the request s that has the highest user inconvenience in r and remove it. Then, we compute the relatedness $\delta(s,q)$ of all other requests q in r. We remove all requests q for which $\delta(s,q) \leq \delta_{\min}$ holds, with $\delta_{\min}$ being a parameter defined with preliminary experiments. Once this is done, we iterate by selecting a new route and iterate until $\lceil nd/100 \rceil$ requests have been removed.

5.3. Repair Operators

We built five operators, which all attempt to reinsert the removed requests by considering all routes in a parallel fashion. In case a repair operator does not manage to reinsert all removed requests, the solution is simply disregarded and the ALNS continues the search from x_{curr}.

Best Insertion. It considers the requests in inverse order with respect to their removal. For each request s, it considers all possible insertion positions, in all routes, and checks whether the insertion would be feasible and how much it would increase the solution value. It then reinserts s in the position leading to the lowest cost increase. The process is repeated until all requests have been reinserted.

Two-Regret Insertion. It works as Best Insertion, but the requests are inserted in non-increasing order of two-regret value. In detail, the operator evaluates for each request the costs of the cheapest insertion position and of the second cheapest insertion position, and it computes the two-regret as the difference between these two costs. It then selects the request of maximum regret and inserts it in the cheapest position. It reiterates, recomputing all regret values at each iteration, until all requests have been reinserted.

Forbidden Insertion. It works as Best Insertion, but disregards the possibility of reinserting a request in the same route which it was removed from.

Perturbation Insertion. It works as Best Insertion, but, any time it computes the cost of inserting a request in a position, it multiplies the cost by a perturbation factor p randomly selected in the interval $[0.8, 1.2]$. The idea, inspired by Ropke and Pisinger (2006), is to add a further level of diversification to the repair process.

Parallel-Set Partitioning Operator. This is the most complex repair method. Starting from the removed requests, the partially destroyed routes, and the airplanes that have not been used, if any, it builds a complete solution by invoking the heuristic of Section 5.1. It invokes the heuristic β times, storing not only the best solution but also all routes from the generated solutions. These routes are then passed to the SP model of Section 4, which is executed for a limited time. The best solution obtained is then returned.

5.4. Adaptive Weight Adjustment

We follow an approach that mimics the classical one of Ropke and Pisinger (2006). At each iteration, we randomly select first a destroy method and next a repair method according to probabilities that depend on the previous results obtained during the

ALNS search. Let M be the set of available destroy methods, $o \in M$ the index of a destroy method, and t the index of the ALNS iteration. At iteration t, each method o is associated with a non-negative weight w^d_{ot}. The weights are used to select a method, according to probabilities $p^d_o = w^d_{ot} / \sum_{q \in M} w^d_{qt}$, for $o \in M$. The same process is applied to select a repair method, for which we produce instead w^r_{ot} weights and p^r_o probabilities.

At the first iteration, all weights are set to the same value, both for destroy and repair, so that they have identical probabilities. Every time a new solution is accepted (as described in Section 5.6 below) the weights of the selected pair of destroy and repair methods are updated. The rationale behind our weight updating is to reward methods that find new improved solutions, with possibly a low computational effort. If a destroy method o has been selected at iteration t and produced an accepted solution, its weight at the next iteration is updated as

$$w^d_{o,t+1} = w^d_{ot} - \text{time}_o/\text{time}_{max} + \Delta_t/\Delta_{max},$$

where time_o is the computing time spent by o, Δ_t is the difference between the cost of the previous and new solutions, computed according to (3), and time_{max} and Δ_{max} are normalization parameters. The same process is used to update the removal weights w^r_{ot}. We note that our weight adjustment mechanism differs from the one by Ropke and Pisinger (2006) because it takes into explicit consideration the computing time spent by an operator. This is due to the fact that our operators require a very different computational effort, with the Parallel-Set Partitioning Operator requiring consistently more time than the other four simpler operators. It is thus important to look not only at the quality of the solutions obtained by an operator, but also at the time needed to obtain it.

5.5. Local Search

The local search approach that we implemented is shown in Algorithm 3. It attempts to improve the input solution by means of four different neighborhoods, invoked one after the other.

Algorithm 3 Local Search

```
1: procedure LS(x_input)
2:     x_LS ← Move(x_input)
3:     x_LS ← Swap(x_LS)
4:     if x_LS = x_input then x_LS ← Inter-move(x_LS)
5:     x_LS ← Time-window manipulation(x_LS)
6:     return x_LS
```

Move. It is an intra-route search that attempts moving a vertex r_i from its current position to another position in route r. For each attempted move, the feasibility of all constraints is checked. In addition, the relocation of a vertex in another position might lead the route to perform two consecutive visits to the same airstrip. In such a case, the two visits are merged into a unique one. We consider a route r at a time, and a vertex in the route at a time, starting from r_2 and continuing until $r_{|r|}$. We attempt each possible relocation, and the one being feasible leading to the highest value, if any, is implemented. The procedure is performed for all routes.

Swap. This procedure is also an intra-route local search. A swap is an intra-route interchange of the positions of two vertices. The process is equivalent to Move, with the only exception that, instead of moving a vertex r_i after another vertex r_j, it swaps r_i with r_j. The swap is checked with respect to feasibility and cost, possibly considering the merging of two visits to the same airstrip into a single visit.

Inter-move. This is the only inter-route search that we implemented. Because it is quite expensive in terms of computing effort, we invoke it only in case the previous intra-route searches failed in finding an improvement. We consider a route and attempt removing from it a pair of consecutive vertices. This is done only if there are no requests that use just one of the two vertices. In such a case, indeed, the removal would create infeasibilities for such requests. We accept, instead, the case in which some requests are picked up in the first vertex and dropped-off in the second. In this case, the requests are also removed from the route. Once the pair of consecutive vertices, and the associated requests, have been removed, we attempt inserting it in all possible positions in the other routes. The insertion being feasible and leading to the highest value reduction, if any, is implemented. The process is iterated until all routes have been scanned.

Time-window manipulation. It is in an intra-route search that evaluates, for each route and for each visited vertex, if it is convenient to increase the time spent by the airplane on the ground. It is just focused on user inconvenience minimization (i.e., the scheduling component of the problem), whereas it does not affect the sequence of visits of the route (routing component). More in detail, at each airstrip the airplane waits for a minimum ground time and possibly for an additional waiting time. This waiting time is kept as small as possible in the constructive heuristic and in the previous local searches to favor the search for feasible solutions. In this local search, we try instead to increase it, by attempting a small subset of possible values. We consider the first vertex in the route and try to increase the time on ground from the original minimal ground time of the airstrip, by attempting all increases of λ minutes each, up to a total of a one-hour increase. The time giving the minimal overall user inconvenience is selected and fixed. We then continue by selecting the next vertex in the route, and again attempting the possible increases of δ minutes in the waiting time. The process is iterated until all vertices have been scanned. We set the value of δ to two minutes on the basis of preliminary experiments.

5.6. Acceptance and Stopping Criteria

We use a simulated annealing acceptance criterion and a geometrical cooling schedule (Delahaye et al. 2019). Given a current solution x, a new solution x' is accepted with probability $P = e^{-(z(x')-z(x))/T_t}$, where $T_t > 0$ is the temperature at iteration t. The temperature starts at $T_1 = T_{\text{start}}$ and is decreased every θ iterations using the expression $T_{t+1} = \alpha T_t$, where $0 < \alpha < 1$ is the cooling rate. Good values for parameters T_{start}, θ and α have been decided on the basis of preliminary computational tests. These tests considered the setting for both the first $iter_{phase1}$ calls to the ALNS method, where we aim at a large diversification, and for the successive calls, where we aim at intensifying the search in promising areas. The value of T_{start} is not fixed as a general input parameter, but we calculate it for each instance, considering the solution of our constructive heuristic. Further details are provided in Section 6.

The ALNS is stopped as soon as one of the following conditions is met: $I_{\max}$ iterations without accepting a new solution have elapsed; a minimum threshold temperature $t_{\min}$ has been reached; or $\Theta_{\max}$ iterations in total, independently from acceptance, have elapsed.

6. Computation Results

In this section, we report the outcome of extensive computational tests that we performed to evaluate the iterated ALNS heuristic. The parameters required by the algorithm were set on the basis of preliminary tests, as follows: in Algorithm 1, $iter_{\max}$=10 and $iter_{phase1}$=$iter_{\max}/2$; for the destroy operators, $[d_{min}, d_{max}]$=$[0.2, 0.6]$ and $\delta_{\min}$=2000; for the Parallel-Set Partitioning repair operator, β is one third of the number of passengers removed from the destroy method (rounded up to the next integer if fractional); for the adaptive weight adjustment, $time_{max}$=20 seconds and Δ_{max}=10000. In terms of acceptance and stopping criteria, during phase 1 we set T_{start}=25000, θ=55, α=0.87, $I_{\max}$=30, $t_{\min}$=150, and $\Theta_{\max}$=20000, whereas in the second phase we set T_{start}=2500, θ=30, α=0.95, $I_{\max}$=80, $t_{\min}$=50, and $\Theta_{\max}$=20.

The algorithm was implemented in C++, and CPLEX 12.9 was used as MILP solver. Computations were made on the computer cluster Beluga from CIRRELT, which uses Intel Gold 6148 Skylake processors running at 2.40 GHz. It is worth noticing that each instance was solved on a single core and that the speed of each core in the grid is similar to that of a standard desktop computer. The tests were performed on a set of real-world instances obtained from the industrial partner. Details on the instances are given in Section 6.1, while in Sections 6.2, 6.3 and 6.4 we contrast our results with those of the company and present a detailed computational analysis.

6.1. Instances

The instance set was created by considering 24 days of activities of the company, as outlined in Table 1. The days are distributed in different times of the year and are consequently characterized by different tourist requests. We divided the instances into three groups: "small" instances have fewer than 150 requests; "medium" instances between 150 and 280 requests; and "large" instances more than 280 requests. Apart from the number of requests, we also provide the number of airstrips, airplane types, and airplanes available. It can be noticed that the test set is quite varied, involving cases having between 91 and 343 requests, between 14 and 21 airstrips, and between 7 and 15 airplanes.

Table 1. Instance characteristics

demand	ID	n	$\|V\|$	$\|F'\|$	$\|F\|$	demand	ID	n	$\|V\|$	$\|F'\|$	$\|F\|$	demand	ID	n	$\|V\|$	$\|F'\|$	$\|F\|$
small	1	91	16	1	11	medium	9	220	19	1	12	large	17	288	18	2	14
small	2	96	16	1	7	medium	10	222	18	2	12	large	18	289	18	2	11
small	3	101	14	1	9	medium	11	226	13	2	11	large	19	292	20	2	14
small	4	110	21	1	10	medium	12	252	16	2	13	large	20	300	16	2	14
small	5	112	19	1	10	medium	13	269	16	2	12	large	21	316	18	2	15
small	6	123	20	1	11	medium	14	271	18	2	12	large	22	320	18	2	15
small	7	125	19	1	10	medium	15	274	19	2	12	large	23	332	21	2	15
small	8	138	21	2	10	medium	16	285	17	1	13	large	24	343	14	2	14

As outlined in Section 2.4, for each instance we tested three cost scenarios. The first one corresponds to a short-term view of the problem, in which the daily cost for using an airplane simply amounts to its mandatory daily fee (c_φ = 30). The second one corresponds instead to a long-term view, in which the airplane cost also includes the daily operating cost of staff salary and amortization ($c_\varphi = 1060$). The third scenario is in the middle between the previous two ($c_\varphi = 545$), and allows us to obtain a sensitivity analysis for the effect of the c_φ parameter on the performance of the optimization method. The values used in (2) to penalize time window violations (ρ_τ) and intermediates stops (ρ_ψ) were set to 1 and 10, respectively. As the time windows are expressed in minutes, with this choice of values an intermediate stop is thus equivalent to 10 minutes of time window violation.

6.2. Results on the short-term scenario and comparison with the company

In Table 2, we present the results we obtained on the short-term scenario, and compare them with the solution implemented by the company. For each instance and each solution, we provide in order: the number of airplanes used (denoted by $|\widetilde{F}|$ in the table); the fuel consumed in liters (fuel); the total distance traveled in kilometers (km); the number of intermediate stops performed (IS); the total time window violation in minutes (TWV); and the objective function values, namey cost (c, computed as in (1)), penalty (u, computed as in (2)), and overall objective (z, computed as in (3) as the sum of c and u). For the iterated ALNS, we also provide the percentage gap from the company solution, computed as $(z_{ALNS} - z_{company})/z_{company} \times 100$, and the overall execution time, in the format h:mm:ss.

Table 2. Comparison with company solutions on the short-term scenario

	company								iterated ALNS									
ID	$\lvert\widetilde{F}\rvert$	fuel	km	IS	TWV	c (1)	u (2)	z (3)	$\lvert\widetilde{F}\rvert$	fuel	km	IS	TWV	c (1)	u (2)	z (3)	gap%	time$_{tot}$
1	11	5852	10888	57	4088	17806.5	5058	22864.5	10	4197	7464	49	1847	12539.5	2337	14876.7	-34.9	0:07:36
2	7	3069	5416	71	2859	9218.7	3889	13107.7	7	2934	5189	80	1744	8772.2	2544	11315.8	-13.7	0:09:32
3	9	3855	7053	63	3112	11750.5	3842	15592.5	9	2909	5136	58	847	8825.9	1427	10253.2	-34.2	0:14:05
4	10	4701	8873	58	1888	14461.4	2748	17209.4	10	4001	7094	32	1067	11968.0	1387	13355.3	-22.4	0:14:33
5	10	4926	9138	76	3120	15085.1	4040	19125.1	10	3495	6156	92	1474	10549.1	2394	12943.0	-32.3	0:14:54
6	11	5449	10154	52	2812	16649.3	3492	20141.3	9	1363	2430	148	2860	4813	4340	9153.3	-54.6	0:27:29
7	10	5626	10262	95	4219	16979.3	5349	22328.3	9	4189	7416	107	4880	12473.8	5950	18423.7	-17.5	0:19:37
8	10	5796	10596	43	5636	17480.3	6206	23686.3	10	3882	6838	69	1958	11658.9	2648	14306.9	-39.6	0:20:36
9	12	5907	10689	114	6195	17842.7	7575	25417.7	12	5593	9879	91	2614	16625.1	3524	20148.9	-20.7	0:54:02
10	12	6782	12515	118	3975	20517.4	5355	25872.4	12	5897	10469	98	2644	17457.9	3624	21081.7	-18.5	1:08:06
11	11	6442	11928	154	8835	19507.8	10575	30082.8	10	5043	8934	91	3309	14906.7	4219	19125.6	-36.4	0:49:34
12	13	7548	13978	112	10673	22919.7	12013	34932.7	13	5495	9707	103	3325	16359.9	4355	20714.8	-40.7	0:57:16
13	12	7250	13240	74	8488	21794.8	9348	31142.8	12	4944	8738	125	3160	14858.8	4409	19268.3	-38.1	1:33:17
14	12	8414	15199	188	10068	25015.8	12128	37143.8	12	5664	10037	139	6276	16790.2	7666	24455.7	-34.2	1:26:22
15	12	7807	14081	190	7311	23230.9	10151	33381.9	12	3480	6180	303	7795	10984.9	10825	21810.3	-34.7	1:56:16
16	13	7128	12852	130	9720	21351.4	11020	32371.4	13	6483	11447	97	2826	19192.3	3796	22987.9	-29.0	1:41:01
17	14	10650	19333	211	8851	31740.4	11101	42841.4	14	7325	12971	173	4893	21756.2	6623	28379.0	-33.8	1:41:28
18	11	7555	13851	171	13254	22804.8	15044	37848.8	11	5849	10322	173	4253	17329.6	5983	23312.2	-38.4	2:06:11
19	14	8168	14442	152	9536	24116.5	11336	35452.5	14	5911	10422	120	3247	17597.4	4447	22044.7	-37.8	1:09:34
20	14	8497	15578	162	16687	25576.1	19007	44583.1	14	6319	11214	126	3235	18863.4	4495	23358.0	-47.6	2:06:46
21	15	8973	16278	184	14951	26898.6	17871	44769.6	15	2873	5112	381	5499	9648.8	9309	18957.7	-57.7	3:27:20
22	15	8914	16443	135	11592	27014.3	13462	40476.3	15	7302	12956	149	5133	21742.7	6623	28365.9	-29.9	2:02:45
23	15	8704	15343	158	16612	25682.9	18692	44374.9	15	7696	13593	206	4179	22917.7	6239	29156.2	-34.3	2:39:58
24	14	8569	15661	196	9355	25763.3	11475	37238.3	14	7609	13474	172	5003	22507.1	6723	29229.7	-21.5	2:54:58
AVG	12.0	6941	12658	124	8077	20883.7	9616	30499.4	11.8	5019	8882	133	3503	15047.5	4829	19876.0	-33.4	1:16:48

From the table, it can be noticed that the iterated ALNS finds solutions that consistently outperform those produced by the company, with percentage gaps ranging from -13.7% to -57.7%, and being -33.4% on average. The solutions by the company are produced manually: the geographical area is divided into two sub-areas, North and South of Tanzania; two employees construct the partial solutions for each area, with the use of Excel files; finally, the two solutions are merged together with some possible adjustments. The process of creating the solution (which we recall is executed the day before) can also be affected by some partial or late information on requests and airplane status, which might cause further adjustments. In this context, the use of the iterated ALNS is well motivated, also due to the fact that the processing times are not excessive, ranging from about seven minutes to about three and a half hours, and being on average around one hour and a quarter.

Strong improvements can be noticed both in the fuel consumption and in the distance travelled, as well as on the two penalizations caused by intermediate stops and time window violations. In terms of airplanes used, the iterated ALNS can reduce this number only for a couple of instances, proving that the fleet is usually well dimensioned for the requests under this service level.

6.3. Results on intermediate and long-term scenarios

In addition to the previous tests, we also attempted to evaluate a long-term scenario and a medium-term scenario, where the daily cost of the airplanes has been increased as previously discussed. The aim is to understand if it is acceptable to reduce the fleet size and how this would affect the service level and the other daily operational costs. The results that we obtained are given in Table 3. The columns have the same meaning as those in Table 2, but we omit the columns with cost c and overall objective function z, as these are affected by the difference in the input costs and cannot be compared among the scenarios.

Table 3. Iterated ALNS results on intermediate and long-term scenarios

	intermediate							long-term						
ID	$\lvert\widetilde{F}\rvert$	fuel	km	IS	TWV	u (2)	time$_{tot}$	$\lvert\widetilde{F}\rvert$	fuel	km	IS	TWV	u (2)	time$_{tot}$
1	9	3961	7055	64	1864	2504	00:06:34	6	3881	6906	68	3920	4600	0:05:38
2	6	3246	5747	77	1602	2372	00:07:47	6	3232	5723	77	2087	2857	0:48:31
3	6	2841	5006	87	2144	3014	00:08:34	6	2909	5129	83	2082	2912	0:07:47
4	7	4015	7105	79	2395	3185	00:09:51	6	3791	6715	84	4398	5238	0:10:53
5	7	3380	5957	102	3075	4095	00:09:35	6	3405	5997	89	4843	5733	0:09:58
6	5	1600	2844	206	3191	5251	00:41:16	6	1700	3018	152	3974	5494	0:23:39
7	7	4335	7674	105	3082	4132	00:11:25	6	4259	7539	115	4580	5730	0:13:22
8	8	4195	7386	90	2640	3540	00:14:46	7	4293	7556	68	5378	6058	0:13:52
9	10	5285	9348	109	5222	6312	00:33:04	9	5400	9535	159	6717	8307	0:37:09
10	11	5910	10496	99	2870	3860	00:36:04	10	5969	10595	89	5053	5943	0:43:07
11	10	4930	8715	166	3476	5136	00:31:14	8	5070	8981	79	7294	8084	0:38:55
12	15	5390	9524	144	4554	5994	00:48:25	11	5556	9814	116	6296	7456	0:58:02
13	10	5100	9004	126	4434	5694	00:54:47	10	5037	8899	87	4886	5756	0:55:15
14	11	5824	10320	132	6402	7722	01:01:48	10	5700	10102	112	7720	8840	0:53:35
15	11	3641	6467	442	6081	10501	01:58:05	9	4318	7669	263	9709	12339	1:26:07
16	13	6827	12052	149	3139	4629	00:46:26	10	6526	11521	145	7416	8866	0:51:35
17	13	7303	12948	187	7139	9009	01:23:35	12	7586	13456	133	9864	11194	0:52:23
18	11	5657	9987	201	5177	7187	01:19:33	10	5800	10255	141	8073	9483	1:03:48
19	11	6381	11252	124	4363	5603	01:05:03	10	6002	10584	103	9343	10373	1:08:44
20	13	6574	11659	120	5635	6835	01:18:39	11	6319	11210	133	8466	9796	1:18:47
21	9	2733	4854	470	8340	13040	01:47:34	8	2633	4686	353	13376	16906	2:49:10
22	14	7356	13035	199	6469	8459	01:34:54	13	7763	13750	128	9245	10525	1:18:35
23	14	7565	13366	200	6118	8118	01:27:03	13	7514	13273	176	8716	10476	1:27:00
24	14	7703	13642	162	7644	9264	02:15:49	12	7810	13829	153	7665	9195	2:01:13
AVG	10.2	5073	8977	160	4461	6061	00:53:00	9.0	5103	9031	129	6713	8007	0:53:25

We can notice that the number $|\widetilde{F}|$ of used airplanes is reduced for all instances when considering the long-term scenario. On average, this number decreases from 11.8 to 9, with a consequent variation of about 24%. This can be imputed to the higher daily usage cost. The side effect is that the routes performed by the airplanes are longer. This can be noticed by considering that the number of airplanes is reduced but at the same time both fuel consumed and traveled distance increase. Another side effect is the increase in the user inconvenience function u. This can be mostly attributed to the increase in the time window violation, that is almost doubled.

The results on the long-term scenario prove that the daily cost for the use of an airplane ($c_\varphi = 30$) has a relevant impact on the solutions. With the intermediate scenario, we can obtain an additional analysis and evaluate what happens when this cost is between the two extremes adopted for the short-term and long-term scenarios.

For the intermediate scenario, the reduction in the number of airplanes used is less evident then the one noticed for the long-term scenario, as the average $|\widetilde{F}|$ value is 10.2 (versus 11.8 for the short-term and 9 for the long-term). For some instances, the difference between the three cases is remarkable, as for instance 1, where $|\widetilde{F}|$ is reduced from 10 to 9 and then to 6. For some other instances, the difference is less obvious. For example, for instances 2, 3 and 13 the long-term solutions use the same number of airplanes than the intermediate ones. However, on these instances the intermediate solutions already used 1, 3 and 2 airplanes less then the short-term ones, respectively. The length of the routes performed in the intermediate scenario is longer than those performed in the short term, as the solutions require to travel, on average, 95 more km. The route lengths are closer to the long-term ones, as the average difference in the traveled distance is just 54 km. The limited difference among these average values is not always obtained on single instances. For example, for instance 4, the distance traveled is 7094 km for the short-term scenario and 7105 km for the intermediate one, but it is then reduced to 6715 km for the long-term, contrarily to what could be expected. This can be imputed to the fact that several equivalent or almost equivalent solutions exist, and the heuristic search might end up in local optima that are quite close with respect to total cost but quite different in terms of traveled distances. For some other cases, instead, the traveled distances increase remarkably. This happens, for example, for instance 22, where the distance increases from 12956 to 13035 and then to 13750 (notably, the traveled distance in the company solution was 16443, so still much higher than those obtained by the heuristic). A similar behavior can be noticed for the fuel consumption, which increases from 5073 (short-term) to 5019 (intermediate) and then to 5103 (long-term). The highest increases in fuel consumption are noticed for those instances where the traveled distances have increased most, as again for instance 22, where the consumption increases from 7302 to 7356 and then to 7763. Another interesting evaluation can be performed on the user inconvenience function u, which is equal to 4829 for the short-term and increases to 6061 for the intermediate and then to 8007 for the long-term. The main component of function u, namely the time-window violation TWV, can change remarkably from the intermediate to the long-term scenario. This behavior is particularly evident for the larger instances, as, for example, for instance 21, where TWV increases from 8340 to 13376. We can conclude that c_φ is a crucial parameter that highly influences the DAFP solutions. The company should evaluate it carefully, so as to avoid an underestimation of the costs, but also an excess in the total inconvenience of the passengers.

Another observation worth noting concerns the number IS of intermediate stops. The company obtained good IS values in their solutions (as also visually depicted in Figure 2 below), proving their attention to this user inconvenience parameter. The iterated ALNS obtained slightly large IS values on the short-term scenario (133 on average, against 124 by the company). The average IS value then raised in the intermediate scenario to 160, and then surprisingly decreased to 129 in the long-term scenario. The decrease from the short-term to the long-term scenario is thus very moderate, with some instances where IS is reduced (as for instances 11, 13–15, 17–19 and 21–24, for which the reduction is of more than 10 stops), and others were instead IS is increased (as for instances 3, 4 and 9, where the increase is of at least 20 stops).

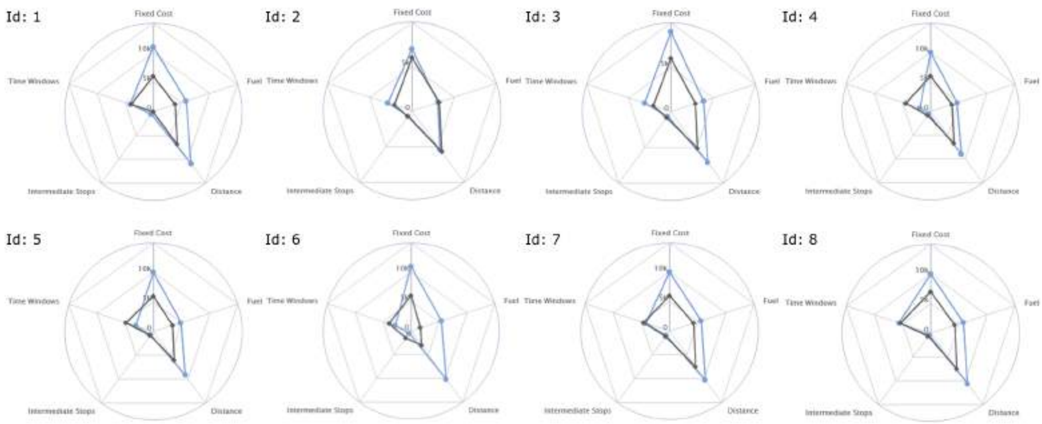
The difference between the intermediate and short-term scenarios is more evident and some sharp increases can be noted (as for instances 6, 11, 15, 16, 21 and 22, which all report an increase of at least 50 stops). This behavior can be imputed to the relative importance of the two user inconvenience components in (2). As noted in Section 6.1, an intermediate stop is equivalent to 10 minutes of time window violation. This implies that an increase in the value of IS can be compensated by a decrease in the TWV value. This can be indeed observed when comparing intermediate and long-term scenarios: the intermediate case has a higher IS value (160 against 132), but also a smaller TWV (6061 against 6713). A full multi-objective assessment of the solutions could provide a full comprehension of the quantitative impact of the different objective function components. Such assessment is out of the scope of this paper, but represents an interesting future research direction.

To further highlight the benefits of using optimization techniques, we performed a deeper instance-by-instance comparison between the company method and the ALNS for the long-term strategy. The outcome is graphically depicted in Figure 2. In this figure, fixed cost, fuel consumption, traveled distance, number of intermediate stops, and time window violation are presented in clock-wise order, enabling to visualize when and by how much the optimization method improves the company solution.

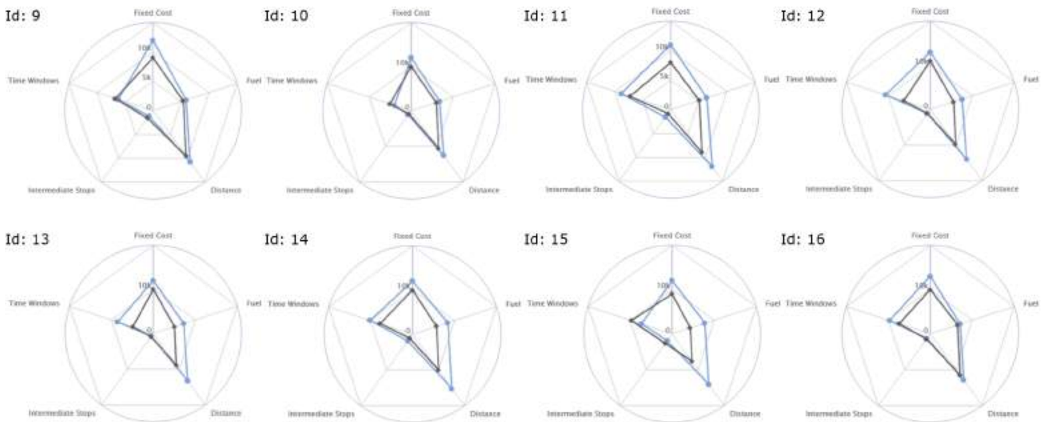
For small-demand instances, there are five cases out of eight in which the solutions found by the ALNS are better than or comparable to the ones found by the company on all the components of the objective function. For the remaining three cases, the company obtains better (i.e., smaller) time window violations. Hence, it appears that being on time is a crucial expectation for the company. This can be imputed to the need to coordinate with ground services in case of early or late arrivals at airstrips located in the middle of the parks (because of safety reasons). Note that to increase the importance of the time window violation component, a user could simply enlarge its weight ρ_τ in (2) before running the ALNS.

A similar but less obvious behavior can be noticed for the medium-demand instances, for which the ALNS solutions are better than or comparable to the company ones for all objective components in seven cases out of eight. For the remaining case, once more the company obtains a smaller time window violation. For the large-demand instances, there is a single case in which the solution by the company is better than the ALNS one for one of the components. In this case, the better value is obtained for the number of intermediate stops. This proves once more the attention of the company to the overall user inconvenience function. Again note that a user could decide to give more importance to the inconvenience function by simply increasing both ρ_ψ and ρ_τ in (2).

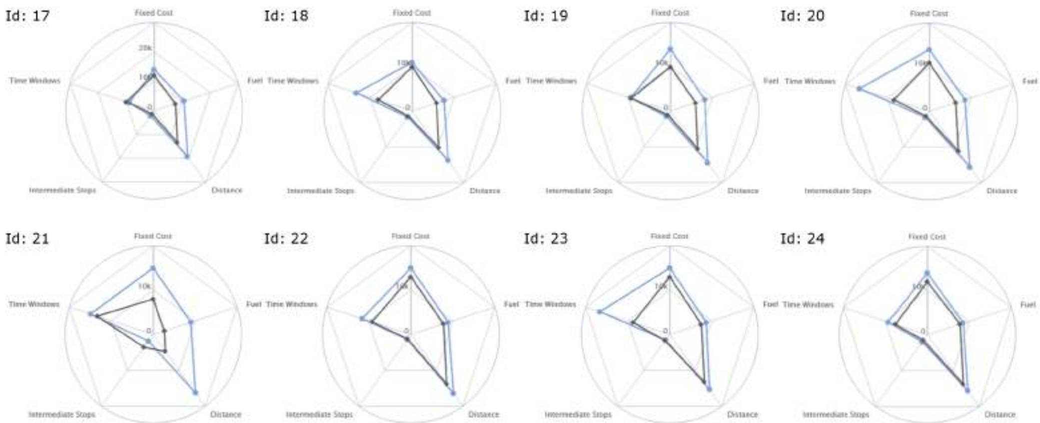
For some instances, the ALNS solution largely dominates the company one on all objective components. This can be especially noticed for medium- and large-demand instances (see, e.g., instances 11, 14, 18 and 20). This proves that the problem gets more complicated when the size of the instance increases and the manual approach used for its solution at the company is not sufficient to achieve optimization. We finally notice that the ALNS solution is never worse than the company one for what concerns the traveled distances. This can be imputed to the fact that the manual approach in use at the company divides the territory into two regions, and produces a sub-optimal solution in which, from time to time, airplanes are sent from one region to the other either empty or with just limited occupancy, to fulfill demands. This behavior, which obviously increases traveled distances, is not noticed in ALNS solutions. We can conclude that the use of optimization techniques is crucial in such a complex problem, and that it is very important to find a good balance between cost



(a) Small-demand instances



(b) Medium-demand instances



(c) High-demand instances

Figure 2. Graphical comparison between long-term iterated ALNS (black lines) and company (blue lines)

and inconvenience components. One might even try to better investigate the role of these components in a multi-objective approach, where, instead of producing a unique solution, a set of a few Pareto-optimal solutions are provided to the decision maker.

6.4. Sensitivity evaluation

With the aim of assessing the impact of the main problem parameters and algorithmic components, we performed three additional tests. In the first test, we executed the iterated ALNS by varying the maximum number of intermediate stops (IS), originally set to 3, by attempting the values 2 and 4. The results are shown in Table 4, where each line provides average values over the entire set of 24 instances. The columns have the same meaning as those in the previous tables, with the exception of a new column, called time$_{incumbent}$, which has been included to present the average time at which the incumbent solution was obtained. We consider the short-term scenario configuration from Section 6.2. As expected, decreasing IS from 3 to 2 has the effect of an increase in the cost c and a decrease in the user inconvenience function u, with an overall slight increase in the resulting objective function z. An opposite behavior can be observed when IS is increased to 4, although no decrease can be observed in the overall function z. We can conclude that 3 is already a large enough value to achieve low-cost routes, and there is no need of accepting 4 or more intermediate stops. The changes in the number of airplanes used are very small (the solutions with at most 4 intermediate stops use just 2 more airplanes than those with 2 stops, considering the entire set of 24 instances). The computing times do not show a correlation with the parameter IS.

Table 4. Analysis of the maximum number of intermediate stops (average results on 24 instances per line, short-term scenario)

max n. IS	$\lvert\widetilde{F}\rvert$	c (1)	u (2)	z (3)	time$_{incumbent}$	time$_{tot}$
2	11.8	15436.6	4777.1	20213.6	00:54:47	00:59:24
3	11.8	15047.5	4828.5	19876.0	01:11:15	01:16:48
4	11.8	15017.3	4949.1	19966.4	00:57:10	01:01:20

In Table 5, we contrast the full iterated ALNS configuration with two reduced configurations, one in which we removed the Inter-move local search operator (Section 5.5), and the other in which we removed the Parallel-Set Partitioning repair operator (Section 5.3). We can notice that both operators have a positive impact on the quality of the solutions obtained by the heuristic. Removing Inter-move leads to a larger increase in the cost c, whereas removing the Parallel-Set Partitioning increases more the user inconvenience. The positive effects of the operators on the solution quality could be noticed on the majority of the attempted instances. No relevant changes can be noticed for the number of used airplanes, whereas the computing effort decreases when removing the operators.

With the the aim of obtaining an evaluation of the different iterated ALNS variants under the same time limit, we performed a further test in which we imposed as additional termination condition a one-hour CPU time limit per instance. We executed once more the three algorithms (namely, full iterated ALNS, iterated ALNS without Inter-move operator and iterated ALNS without Parallel-Set Partitioning operator) on all instances. For the sake of conciseness, we do not report a new table but comment the results we obtained.

The full iterated ALNS used an average total time of 37 minutes and 11 seconds, against 38 minutes and 15 seconds for the ALNS without Inter-move operator, and 35 minutes and 36 seconds for the ALNS without Parallel-Set Partitioning, so the total efforts are very similar. The time required to reach the incumbent solution was on average 34 minutes and 14 seconds for the full algorithm, against 36 minutes and 2 seconds for the case without Inter-move and 34 minutes and 54 seconds for the case without Parallel-Set Partitioning, so again very similar values. In terms of costs, the full ALNS obtained average values for c, u and z being equal to, respectively, 15058.2, 4840 and 19898.7. The ALNS without Inter-move obtained, instead, 15324.1, 4826 and 20149.9, whereas the ALNS without Parallel-Set Partitioning 15151.8, 5075 and 20226.9, respectively. We can thus observe that the removal of the Inter-move operator led to an increase of 1.65% in the solution cost (with a 0.62% increase in the operating costs and 4.85% increase in the user inconvenience) and the removal of the Parallel-Set Partitioning operator to an increase of 1.26% in the solution cost (with a 1.77% increase in the operating costs and 0.3% decrease in the user inconvenience). This additional test further confirms the efficacy of the new operators in improving the performance of the iterated ALNS.

Table 5. Evaluation of the new operators (average results on 24 instances per line, short-term scenario)

configuration	$\lvert\widetilde{F}\rvert$	c (1)	u (2)	z (3)	time$_{incumbent}$	time$_{tot}$
full iterated ALNS	11.8	15047.5	4828.5	19876.0	01:11:15	01:16:48
no Inter-move	11.8	15165.2	4895.6	20060.7	00:54:54	01:02:23
no Parallel-Set Partitioning	11.8	15133.8	4915.7	20049.4	00:53:10	00:56:22

With the aim of determining the best balance between the time spent in the heuristic search and in the MILP model solution by the solver, we performed an additional computational analysis in which we attempted different values of the number of calls to Algorithm 2 inside the iterated ALNS. This has been obtained by changing the value of $iter_{\max}$ (see step 5 of Algorithm 1), which also represents the number of calls to the set partitioning model (step 19 of Algorithm 2). We performed these tests on the long-term scenario configuration already studied in Table 3. The results that we obtained are summarized in Table 6. We can notice that the attempt with $iter_{\max}$=10 (i.e., the value we adopted for all our previous experiments) gives slightly better results than the other attempts, providing lower cost and user inconvenience values. The improvements are quite small with respect to the values obtained with six and 12 iterations, but much better with respect to those obtained with just one or two attempts. This proves that a good number of calls to the set partitioning model is beneficial for the overall algorithm, and that, when this value is sufficiently large, the algorithm becomes robust.

Table 6. Analysis for different calls to the set partitioning model (average results on 24 instances per line, long-term scenario)

$iter_{\max}$	$\lvert\widetilde{F}\rvert$	c (1)	u (2)	z (3)	time$_{incumbent}$	time$_{tot}$
1	9.0	32363.4	9655	42018.5	00:43:25	00:53:19
2	8.8	31255.5	9596	40851.9	00:41:08	00:54:09
6	9.1	31441.8	8391	39833.0	00:48:51	00:53:46
10	9.0	31018.0	8007	39025.0	00:47:10	00:53:25
12	9.0	31064.5	8008	39072.7	00:52:23	00:55:12

7. Conclusions

In this paper, we studied the Dial-A-Flight operations of one of the major Safari airline companies in Tanzania. The problem they face, denoted DAFPAS, is very challenging because it combines a heterogeneous aircraft fleet, multiple depots, flexible time windows, different operational costs, and the need to provide a good service level. The service level is measured by the number of intermediate stops that passengers undertake during transportation, and the possible violation of the flexible time window constraints. Another complicating issue in the problem originates from the fact that refueling is possible only at a limited number of airstrips.

We solved the DAFPAS heuristically with an iterated ALNS algorithm. Consistently with the literature, the ALNS proved to be effective in dealing with large size instances, finding solutions that were consistently better than those produced manually at the company. Local search and a set partitioning model helped improve the performance of the ALNS. In particular, it was shown that it is better to invoke the set partitioning model many times, with a shorter time limit, instead of just once with a longer limit. Attempting the exact solution of the set partitioning model, by means of a dedicated column generation algorithm, could represent an interesting future contribution aimed at providing optimal or near-optimal solutions for the problem.

An interesting future research direction is to consider the planning of the itineraries for multiple consecutive days, so as to find the best airstrips where to stop during the night and start at the next morning. That would require modifying the heuristic algorithm, both for what concerns ALNS, local search and set partitioning components, by considering the fact that routes selected for a given day should be connected with the routes in the next day. Such an approach could be employed within a rolling horizon framework. Within this context, it would be relevant to plan also the pilots' shifts, in a possible attempt to optimize the crew rostering component of the resulting problem.

In our case study, the ground time was always large enough to allow for a full refueling of an airplane. In addition to that, we assumed the fuel costs to be identical at all airstrips. These conditions might not be satisfied in other Dial-A-Flight problems, for example when the airplanes are large and travel across different countries. In such cases, the underlying scheduling problem would be more complicated, both in terms of user inconvenience function and cost, and the development of tailored solution algorithms would be important to achieve good quality solutions.

In the sensitivity analyses performed by attempting different cost scenarios, we experienced relevant variations in one or more components of the objective function, as, e.g., the number of intermediate stops for the passengers. Indeed, the iterated ALNS found on all instances and scenarios solutions whose overall objective function value was better than the one found by the company. However, in some instances the company was able to obtain better quality of service indicators, with less time window violation or a reduced number of intermediate stops. To better assess the impact of each objective function component, a multi-objective approach could be developed. It would be necessary, in such a case, to return not a single incumbent solution, but a set of non-dominated solutions. As the task would be even more difficult than solving the single-objective problem, it is envisaged to maintain the use of a metaheuristic algorithm, possibly with an extended time limit with respect to the ones allowed in our work. The idea, adopted in other Dial-A-Ride-Problems with conflicting objective functions (see, e.g., Ho et al. 2018) can be beneficial at a practical level only if the decision maker is provided with some evaluation/visualization tools to quickly assess the positive and negative aspects of a solution with respect to another.

Another interesting research avenue concerns the opportunity to issue low-cost last minute fares, so as to fill remaining seats at the planned trips, or obtain a better use for trips that are only meant at relocating the aircraft to meet successive requests. Alternative means of transport could also be taken into account. Indeed, for itineraries of limited distance, the tourists can also be offered to move on road, as in traditional ground safaris.

Acknowledgement(s)

Computations were made on the supercomputer Beluga from CIRRELT, managed by Calcul Québec and Compute Canada. The operation of the supercomputer is funded by the Canada Foundation for Innovation, Ministre de l'Économie et de l'Innovation, and Fonds de recherche Nature et technologies - Québec. We are grateful to Enrico Tognoni for providing us with data and several interesting insights on air safaris.

References

C. Barnhart, P. Belobaba, and A.R. Odoni. Applications of operations research in the air transport industry. *Transportation Science*, 37(4):368–391, 2003.

M. Battarra, J.-F. Cordeau, and M. Iori. Pickup and delivery problems for goods transportation. In P. Toth and D. Vigo, editors, *Vehicle Routing: Problems, Methods, and Applications*, MOS-SIAM Series on Optimization, pages 161–192. SIAM, 2014.

T. Bektaş, E. Demir, and G. Laporte. Green vehicle routing. In *Green Transportation Logistics*, pages 243–265. Springer, 2016.

V. Cacchiani and J.-J. Salazar-González. Heuristic approaches for flight retiming in an integrated airline scheduling problem of a regional carrier. *Omega*, 91:102028, 2020.

Civil Aviation, Tanzania. The civil aviation (operation of aircraft) regulations. Technical report, available at `https://www.tcaa.go.tz/` (last accessed March 2021), 2017.

J.-F. Cordeau and G. Laporte. A tabu search heuristic for the static multi-vehicle dial-a-ride problem. *Transportation Research Part B: Methodological*, 37(6):579–594, 2003.

J.-F. Cordeau and G. Laporte. The dial-a-ride problem: models and algorithms. *Annals of Operations Research*, 153(1):29–46, 2007.

J.-F. Cordeau, G. Laporte, J.Y. Potvin, and M.W.P. Savelsbergh. Chapter 7 in Transportation on Demand. *Handbooks in Operations Research and Management Science*, 14:429–466, 2007.

R. de Alvarenga Rosa, A.M. Machado, G.M. Ribeiro, and G.R. Mauri. A mathematical model and a clustering search metaheuristic for planning the helicopter transportation of employees to the production platforms of oil and gas. *Computers & Industrial Engineering*, 101:303–312, 2016.

D. Delahaye, S. Chaimatanan, and M. Mongeau. Simulated annealing: From basics to applications. In M. Gendreau and J.-Y. Potvin, editors, *Handbook of Metaheuristics*, volume 272 of *International Series in Operations Research & Management Science*, pages 1–35. Springer, 2019.

M. Dell'Amico, J.C. Díaz Díaz, G. Hasle, and M. Iori. An Adaptive Iterated Local Search for the Mixed Capacitated General Routing Problem. *Transportation Science*, 50(4):1223–1238, 2016.

G. Desaulniers, J. Desrosiers, Y. Dumas, M. Solomon, and F. Soumis. Daily aircraft routing and scheduling. *Management Science*, 43(6):841–855, 1997.

K. Doerner and J.-J. Salazar-González. Pickup and delivery routing problems for people transportation. In P. Toth and D. Vigo, editors, *Vehicle Routing: Problems, Methods, and Applications*, MOS-SIAM Series on Optimization, pages 193–212. SIAM, 2014.

F.G. Engineer, G.L. Nemhauser, and Martin W.P. Savelsbergh. Dynamic programming-based column generation on time-expanded networks: Application to the dial-a-flight problem. *INFORMS Journal on Computing*, 23(1):105–119, 2011.

D. Espinoza, R. Garcia, M. Goycoolea, G.L. Nemhauser, and M. W.P. Savelsbergh. Per-seat, on-demand air transportation Part I: problem description and an integer multicommodity flow model. *Transportation Science*, 42(3):263–278, 2008a.

D. Espinoza, R. Garcia, M. Goycoolea, G.L. Nemhauser, and M. W.P. Savelsbergh. Per-seat, on-demand air transportation Part II: Parallel local search. *Transportation Science*, 42(3): 279–291, 2008b.

K. Fagerholt, B.A. Foss, and O.J. Horgen. A decision support model for establishing an air taxi service: a case study. *Journal of The Operational Research Society*, 60(9):1173–1182, 2009.

M.T. Fiala Timlin and W.R. Pulleyblank. Precedence constrained routing and helicopter scheduling: heuristic design. *Interfaces*, 22(3):100–111, 1992.

T. Gschwind and M. Drexl. Adaptive Large Neighborhood Search with a Constant-Time Feasibility Test for the Dial-a-Ride Problem. *Transportation Science*, 53(2):480–491, 2019.

N.S.S. Hermeto, V.J.M. Ferreira Filho, and L. Bahiense. Logistics network planning for offshore air transport of oil rig crews. *Computers & Industrial Engineering*, 75:41–54, 2014.

Sin C Ho, WY Szeto, Yong-Hong Kuo, Janny MY Leung, Matthew Petering, and Terence WH Tou. A survey of dial-a-ride problems: Literature review and recent developments. *Transportation Research Part B: Methodological*, 111:395–421, 2018.

T.E. Kannon, S.G. Nurre, B.J. Lunday, and R.R. Hill. The aircraft routing problem with refueling. *Optimization Letters*, 9(8):1609–1624, 2015.

P. Keskinocak and S. Tayur. Scheduling of time-shared jet aircraft. *Transportation Science*, 32(3):277–294, 1998.

D. Klabjan. Large-scale models in the airline industry. In G. Desaulniers, J. Desrosiers, and M.M. Solomon, editors, *Column Generation*, pages 163–195. Springer, Boston, MA, 2005.

E. Lacasse-Guay, G. Desaulniers, and F. Soumis. Aircraft routing under different business processes. *Journal of Air Transport Management*, 16(5):258–263, 2010.

C. Martin, D. Jones, and P. Keskinocak. Optimizing on-demand aircraft schedules for fractional aircraft operators. *Interfaces*, 33(5):22–35, 2003.

M.A. Masmoudi, K. Braekers, M. Masmoudi, and A. Dammak. A hybrid Genetic Algorithm for the Heterogeneous Dial-A-Ride Problem. *Computers & Operations Research*, 81:1–13, 2017.

R. Masson, F. Lehuédé, and O. Péton. An Adaptive Large Neighborhood Search for the Pickup and Delivery Problem with Transfers. *Transportation Science*, 47(3):344–355, 2013.

F. Menezes, O. Porto, M.L. Reis, L. Moreno, M. Poggi de Aragão, E. Uchoa, H. Abeledo, and N. Carvalho do Nascimento. Optimizing helicopter transport of oil rig crews at petrobras. *Interfaces*, 40(5):408–416, 2010.

P. Munari and A. Alvarez. Aircraft routing for on-demand air transportation with service upgrade and maintenance events: Compact model and case study. *Journal of Air Transport Management*, 75:75–84, 2019.

National Bureau of Statistics, Tanzania. Tanzania in Figures. Technical report, available at https://www.nbs.go.tz/index.php/en/tanzania-in-figures/641-tanzania-in-figures-2020 (last accessed December 2021), 2020.

A. Parmentier and F. Meunier. Aircraft routing and crew pairing: Updated algorithms at Air France. *Omega*, 93:102073, 2020.

S.N. Parragh, K.F. Doerner, and R.F. Hartl. Variable neighborhood search for the dial-a-ride problem. *Computers & Operations Research*, 37(6):1129–1138, 2010.

D. Pisinger and S. Ropke. Large neighborhood search. In M. Gendreau and J.Y. Potvin, editors, *Handbook of metaheuristics*, volume 146 of *International Series in Operations Research & Management Science*, pages 399–419. Springer, Boston, MA, 2010.

F. Qian, I. Gribkovskaia, G. Laporte, and Halskau sr Ø. Passenger and pilot risk minimization in offshore helicopter transportation. *Omega*, 40(5):584–593, 2012.

D. Ronen. Scheduling charter aircraft. *Journal of the Operational Research Society*, 51(3): 258–262, 2000.

S. Ropke and D. Pisinger. An Adaptive Large Neighborhood Search Heuristic for the Pickup and Delivery Problem with Time Windows. *Transportation Science*, 40(4):455–472, 2006.

P. Shaw. A new local search algorithm providing high quality solutions to vehicle routing problems. Technical report, Dept. of Computer Science, University of Strathclyde, Glasgow, Scotland, UK, 1997.

A. Subramanian, E. Uchoa, and L.S. Ochi. A hybrid algorithm for a class of vehicle routing problems. *Computers & Operations Research*, 40(10):2519–2531, 2013.

F.M. Van der Zwan, K. Wils, and S.S.A. Ghijs. Development of an aircraft routing system for an air taxi operator. In *Aeronautics and Astronautics*. IntechOpen, 2011.

J. Wensveen. *Air transportation: A management perspective*. Routledge, 2016.

A. Yamani, T.J. Hodgson, and L.A. Martin-Vega. Single aircraft mid-air refueling using spherical distances. *Operations Research*, 38(5):792–800, 1990.

W. Yang, I.Z. Karaesmen, P. Keskinocak, and S. Tayur. Aircraft and crew scheduling for fractional ownership programs. *Annals of Operations Research*, 159(1):415–431, 2008.

Y. Yao, Ö. Ergun, E. Johnson, W. Schultz, and J.M. Singleton. Strategic planning in fractional aircraft ownership programs. *European Journal of Operational Research*, 189(2):526–539, 2008.

Y. Yuan and A. Mehrez. Refueling strategies to maximize the operational range of a nonidentical vehicle fleet. *European Journal of Operational Research*, 83(1):167–181, 1995.