A constraint programming model and a hybrid iterated local search algorithm for solving an aircraft recovery problem in the oil and gas industry

Mateus Martin[a,*], Aldair Alvarez[b], Jonathan De La Vega[b], Reinaldo Morabito[b]

[a]*Departamento de Engenharia de Produção, Universidade Federal Fluminense, Rua Domingos Silvério 135, 25650-050, Petrópolis-RJ, Brazil*
[b]*Departamento de Engenharia de Produção, Universidade Federal de São Carlos, Via Washington Luiz km 235, 13565-905, São Carlos-SP, Brazil*

Abstract

In this paper, we address a challenging problem faced by a Brazilian oil and gas company regarding the rescheduling of helicopter flights from an onshore airport to maritime units, crucial for transporting company employees. The problem arises due to unforeseen events like bad weather or mechanical failures, leading to delays or postponements in the original flight schedules, disrupting the operation of maritime units and employee shift scheduling. To model and solve the problem, we propose a constraint programming (CP) model aimed at optimizing daily flight scheduling with minimal delay and helicopter usage, considering various constraints like rescheduling priorities and time windows. We also develop a hybrid iterated local search algorithm to handle larger instances of the problem for the case when a general-purpose CP solver may not be available. Our approaches, evaluated using real-world data, demonstrate their effectiveness in solving short-term flight rescheduling problems in the context of the oil and gas industry, in comparison to exact and heuristic approaches from the literature.

Keywords: Short-term flight rescheduling, aircraft recovery problem, helicopters and flights re-assignments, constraint programming, iterated local search.

1. Introduction

We address an aircraft recovery problem with priority distinction between flights, fleet, and delays (ARP-PD). This ARP-PD entails a rescheduling problem involving helicopter flights transporting personnel from an onshore airport to various offshore units (maritime units), exemplified by offshore oil rigs and gas-producing platforms, drawing from a real case in a Brazilian oil and gas company (Vieira et al., 2021). These flights primarily serve to transport crews beginning or ending duties at each maritime unit, with the company establishing weekly flight schedules specifying departure times and destinations. Each flight typically comprises a round trip between the airport and a maritime unit, ferrying pre-booked personnel to and from their assigned stations. Despite

*Corresponding author.
Email addresses: mpmartin@id.uff.br (Mateus Martin), aldair.alvarez@outlook.com (Aldair Alvarez), jdlvmartinez@gmail.com (Jonathan De La Vega), morabito@dep.ufscar.br (Reinaldo Morabito)

initial scheduling, deviations occur due to unforeseen events like adverse weather, passenger delays, and helicopter malfunctions, necessitating on-the-fly adjustments or flight postponements to the following day, termed transferred flights. Effective rescheduling is crucial, as delays incur penalties and impact operational efficiency. The complexity of this ARP-PD stems from numerous factors – this oil and gas company ranks among Brazil's top four companies in flight volume – with dozens of maritime units, daily flights, and available helicopters. Similar challenges are prevalent among oil companies globally, from the North Sea to the Gulf of Mexico and beyond. In this study, we assume deterministic parameters, simplifying the problem as one of assigning round-trip flights to helicopters, sequencing their daily journeys, and coordinating flights at airport runways and maritime unit heliports.

There are literature review studies that have compiled and analyzed the knowledge developed on aircraft recovery problems since the pioneering study in Teodorović & Guberinić (1984). Some examples include Filar et al. (2001), Kohl et al. (2007), Clausen et al. (2010), Hassan et al. (2021), and Santana et al. (2023). All of these studies addressed the problems comprehensively, focusing on airline disruption management and addressing the main factors that influence the instability of flight control operations. Regarding the problem of aircraft recovery in the context of oil and gas companies, Vieira et al. (2021) and De La Vega et al. (2022a,b) addressed the transport of passengers from onshore airports to offshore units using helicopters.

Vieira et al. (2021) proposed two Mixed Integer Linear Programming (MILP) models, one based on a flow network and another based on events. Several characteristics were considered in the problem, for example, the simultaneous incorporation of flights with different rescheduling priorities together with the possibility of using spare helicopters, the existence of a single runway for takeoff and landing at the airport, strict operational rules relating to daily flight schedules, limitations on the number and type of helicopters available for some flights, among others. Furthermore, the authors also proposed a constructive heuristic to solve this ARP-PD. The present study is closely related to this study. Other studies related to Vieira et al. (2021) appear in De La Vega et al. (2022a,b). De La Vega et al. (2022a) also considered subsets of mandatory flights and restrictions related to different helicopter rental contracts and mandatory crew lunch stops. The study presented model-based solution approaches that investigate discrete-time MILP models to handle larger problem instances and explored ad-hoc constructive and improvement heuristics. De La Vega et al. (2022b) extended the continuous-time MILP models and the constructive heuristic of Vieira et al. (2021) to deal with multiple airports and transfers of delayed flights between these airports to maritime units.

The main contribution presented in this paper is two-fold. First, we propose a constraint programming (CP) model that fully represents the ARP-PD of Vieira et al. (2021). Next, we develop an iterated local search algorithm in conjunction with a column generation Integer Linear Programming (ILP) model to achieve satisfactory quality solutions for larger problem instances when a general-purpose CP solver may not be available. Our approaches, evaluated using real-world data, demonstrate their effectiveness in solving short-term flight rescheduling problems in the context of the oil and gas industry, in comparison to exact and heuristic approaches found in the literature. Additionally, we aim to contribute to the development of diverse solution strategies for a practical operational research problem. These exact modeling strategies and heuristics contribute to the characterization and understanding of the problem, and may motivate the development of new approaches in future research for related problems in other companies in the oil and gas sector around the world.

The remainder of this paper is organized as follows. We describe the ARP-PD in details in Section 2. Next, we introduce the proposed CP model and hybrid iterated local search algorithm for the ARP-PD in Sections 3 and 4, respectively. An ILP model based on column generation for the problem is presented in the Appendix to be used as a one-shot post-optimization step after the proposed meta-heuristic algorithm. In Section 5, we present computational results evaluating our approaches using real-life data from a Brazilian oil company. Finally, we conclude with remarks in Section 6.

2. Description of the problem

According to Vieira et al. (2021), the ARP-PD is motivated by a real-life short-term rescheduling operation of helicopter flights that transport personnel to and from maritime units in the context of a Brazilian oil and gas company. At the beginning of the week, the company programmers schedule helicopter flights for each day of the week. Each flight is a round-trip flight (airport-maritime unit-airport) that takes work teams to and from the maritime units, taking into account the work teams' shifts. The transport team will carry out these flights in the absence of unexpected events. However, in practice, unexpected events occur mainly associated with weather conditions which may force previously scheduled flights to be delayed or transferred to the following day. To cope with transferred flights, at the beginning of each working day, the company programmers revise and adjust the day timetable to include the flights transferred from the previous day or even any day before that on the current day. The company programmers treat the optimization decisions of this operation as an aircraft recovery problem while prioritizing between types of flights, types of helicopters used, and the delays of the flights.

To formally characterize the ARP-PD, we first define the following sets:

$P = \{1, \ldots, \overline{P}\}$ set of maritime units (i.e., offshore platforms are the flight destinations);
$I = \{1, \ldots, \overline{I}\}$ set of flights;
$H = \{1, \ldots, \overline{H}\}$ set of helicopters.

In what follows, we describe the characteristics and requirements of the ARP-PD concerning the airport, maritime units, helicopters, and flights:

- **Airport**. There is an onshore airport with a single runway. The airport's opening hours follow a time window $[tw^a, tw^b]$ depending on sunlight (e.g., 7:00 am to 6:00 pm). Therefore, all flights must depart and land within this time window. For safety reasons, there is a precedence constraint between two consecutive flights, that is, the departure time of two flights in a row from the airport must respect a minimum time interval t^a (e.g., 5 min).

- **Maritime units**. Each maritime unit $p \in P$ has a single heliport. In other words, each maritime unit p can receive at most one helicopter on the ground per unit of time. The landing, boarding and unboarding of the work team, and take-off operation of a helicopter at a maritime unit p takes a minimum time interval t_i^u (e.g., 15 min) that blocks this unit from receiving other helicopters. Thus, we should guarantee a precedence constraint, that is, the difference between the departure times of two flights $i, j \in I$, $i \neq j$, from the airport towards the same maritime unit p must be greater than or equal to the value of t_i^u if flight i departures before flight j or to the value of t_j^u if flight j departures before flight i.

- **Helicopters**. There are three types of helicopters:

$H_n \subseteq H$ set of helicopters in the normal set;
$H_p \subseteq H$ set of helicopters in the pool set;
$H_s \subseteq H$ set of helicopters in the spot set.

We observe that $H = H_n \cup H_p \cup H_s$ and the intersection between any two of these three sets is empty (i.e., $H_n \cap H_p = H_n \cap H_s = H_p \cap H_s = \emptyset$). The oil and gas company has a long-term contract for the fleet of helicopters. Each normal helicopter $h \in H_n$ is initially assigned to a flight in the current day timetable. Each pool helicopter $h \in H_p$ is readily available at the airport under additional costs, as it is not previously assigned to any flight. Each spot helicopter $h \in H_s$ is not readily available at the airport, which incurs higher additional costs to be used. We note that a helicopter $h \in H$ may perform several flights in a day; however, there is a minimal time interval *tat* on the ground at the airport between two flights in a row of the same helicopter due to an inspection procedure for safety reasons. Furthermore, the helicopter fleet is heterogeneous in terms of personnel transport capacity and autonomy to cover distances. Thus, we define:

$H_i \subseteq H$ set of helicopters able to perform flight $i \in I$.

- **Flights**. There are four types of flights:

  $I_0 \subseteq I$ set of table flights;
  $I_1 \subseteq I$ set of transferred flights from the previous day;
  $I_2 \subseteq I$ set of transferred flights from two or more days before;
  $I_E \subseteq I$ set of entourage flights.

  We observe that $I = I_0 \cup I_1 \cup I_2 \cup I_E$ and the intersection between any two of these four sets is empty. Each table flight $i \in I_0$ is previously scheduled in the current day timetable. A transferred flight $i \in I_1 \cup I_2$ is a flight that needs to be rescheduled on the current day. As a general rule, a flight $i \in I_0 \cup I_1 \cup I_2$ takes a group of passengers to the destination maritime unit and brings back a distinct group of passengers, depending on the work teams' shifts of that maritime unit. Finally, an entourage flight $i \in I_E$ is a special type of flight associated with the transport of personnel from the oil and gas company's management team or the regulatory agency.

  There are two organizational policies regarding flights that impose precedence constraints. The first one regards that the departure time of a transferred flight $i \in I_1 \cup I_2$ must precede the departure time of a table flight $j \in I_0$ if both flights have the same maritime unit $p \in P$ as a destination. The second one concerns that a maritime unit/helicopter used must be blocked from receiving/performing other flights after the completion of an entourage flight $i \in I_E$, given that unforeseen work situations often extend the time of an entourage flight on the ground at its destination maritime unit. In addition, we define:

  $I_h \subseteq I$ set of flights that can be assigned to helicopter $h \in H$;
  $I_p \subseteq I$ set of flights with maritime unit $p \in P$ as destination.

  Each flight $i \in I$ has a scheduled departure time r_i and a duration t_i (total time of the airport-maritime unit-airport operation). However, any flight $i \in I$ may be delayed on the current day seeking to cope with the inclusion of any transferred flight $j \in I_1 \cup I_2$ into the day timetable. There are two types for delaying a flight (a short delay/type I and a longer delay/type II) that respect the following parameters:

d^I_{max} maximum delay of type I allowed for the departure time of a flight $i \in I_E \cup I_0$ (e.g., 15 min);
d^{II}_{max} maximum delay of type II allowed for the departure time of a flight $i \in I_E \cup I_0$ (e.g., 240 min).

We note that $d^{II}_{max} > d^I_{max}$. Any flight $i \in I_E \cup I_0$ with a departure time delayed more than d^{II}_{max} is not performed on the current day, that is, it will be transferred for the next day. Finally, each flight $i \in I_E \cup I_0$ of the day timetable has a helicopter $s_i \in H_i$ initially assigned to it; however, due to the rescheduling of flights, another helicopter $h \in H_i$, $h \neq s_i$, may be assigned to perform this flight.

The objective of the ARP-PD consists of a weighted sum with eleven different penalties w_q, $q \in \{1, \ldots, 11\}$, defined by the company programmers that prioritizes between types of flights, types of helicopters used, and the delays of the flights. They are briefly discussed in what follows (for more details of these priorities, please see Vieira et al. (2021)):

- types of flights: there are four different penalties for not scheduling each entourage flight $i \in I_E$ (w_1), transferred flight $i \in I_2$ of two or more days (w_2), transferred flight $i \in I_1$ of the previous day (w_3), and table flight $i \in I_0$ (w_4), with $w_1 \geq w_2 \geq w_3 \geq w_4$;
- types of helicopters used: there are three different penalties for using each helicopter type, that is, a spot helicopter $h \in I_s$ (w_5), a pool helicopter $h \in I_p$ (w_6), and normal helicopter $h \in I_n$ (w_7) on the current day, with $w_5 \geq w_6 \geq w_7$;
- changes in the previous assignment of helicopters: there is a penalty for not using the initially assigned helicopter s_i to perform flight $i \in I_E \cup I_0$ (w_8);
- delay of the flights: there are three different penalties when delaying a flight, where two of them concern the delays of types II (w_9) and I (w_{10}) for each flight $i \in I_E \cup I_0$ and one of them concern the difference between the departure and release times (i.e., the value of the delay) of each flight $i \in I$ (w_{11}), with $w_9 \geq w_{10} \geq w_{11}$.

3. Constraint programming model

In this section, we propose a constraint programming (CP) model for the ARP-PD relying on interval variables and interval sequence variables. We observe that an interval variable is a decision variable of the CP paradigm that represents the execution of an activity over a time interval. An interval variable is characterized by an earliest/latest start time, an earliest/latest end time, and a minimum/maximum duration of the activity; however, the real values that this variable will assume for its start/end times and duration are unknown to the problem. Moreover, one can define if this interval variable must be present (i.e., the activity must be executed) or optional (i.e., the activity will be executed or absent) in the solution. We next model the departure/start time, arrival/end time, and duration of each flight $i \in I$ as well as the working time of each helicopter $h \in H$ as interval variables. We also observe that an interval sequence variable α on a set of interval variables $\{\beta_1, \beta_2, \ldots, \beta_m\}$ (or shortly $\{\beta_k\}_{k=1,\ldots,m}$) represents a decision variable of the CP paradigm whose possible values are all the permutations of this set of interval variables. Thus, we next model the sequence of flights performed by the same helicopter $h \in H$ as an interval sequence variable.

The proposed CP model has seven types of decision variables which are defined as follows:

x_i an optional interval variable representing the execution of flight $i \in I$ with a possible start r_i, a possible end tw_b, and a fixed duration t_i;

y_{ih} an optional interval variable representing the execution of flight $i \in I$ to be performed by helicopter $h \in H_i$ with a possible start r_i, a possible end tw_b, and a fixed duration t_i;

z_h an optional interval variable representing the use of helicopter $h \in H$ with a possible start tw_a and a possible end tw_b;

π_h interval sequence variable representing the sequence of flights (i.e., a permutation) performed by helicopter $h \in H$;

d_i integer variable representing the delay of flight $i \in I$, that is, the difference between its departure/start time and release time r_i;

b_i^I binary variable which equals 1 if flight $i \in I_E \cup I_0$ has a delay of type I ($0 < d_i \leq d_{max}^I$), and 0 otherwise;

b_i^{II} binary variable which equals 1 if flight $i \in I_E \cup I_0$ has a delay of type II ($d_{max}^I < d_i \leq d_{max}^{II}$), and 0 otherwise.

For the sake of explanation, we present the proposed CP model in blocks of constraints with their respective explanation in what follows using the OPL modeling language (Van Hentenryck, 1999).

Similar to Vieira et al. (2021), the objective function (1) is based on the policy of the oil and gas company to deal with flight recovery. Notice that the first eight terms of the objective function use the predicate `presence` over an interval variable; this predicate will return one if such an interval variable is present in the solution (e.g., the flight is performed/the helicopter is used), and zero otherwise (e.g., the flight is not performed/the helicopter is not used). In this sense, the first four terms penalize the number of flights not performed on the current day concerning entourage flights, transferred flights of two or more days, transferred flights from the previous day, and table flights, respectively. The following three terms penalize the number of helicopters used concerning the spot, pool, and normal sets, respectively. The eighth term penalizes the number of flights $i \in I_E \cup I_0$ not using the respective initially assigned helicopter s_i for their operations. The ninth and tenth terms penalize the number of delays of type II and type I, respectively. Finally, the eleventh term penalizes the total sum of the delay of the flights.

$$
\begin{aligned}
\textbf{Min} \quad & w_1\left(|I_E| - \sum_{i \in I_E} \texttt{presence}(x_i)\right) + w_2\left(|I_2| - \sum_{i \in I_2} \texttt{presence}(x_i)\right) \\
& + w_3\left(|I_1| - \sum_{i \in I_1} \texttt{presence}(x_i)\right) + w_4\left(|I_0| - \sum_{i \in I_0} \texttt{presence}(x_i)\right) \\
& + w_5 \sum_{h \in H_s} \texttt{presence}(z_h) + w_6 \sum_{h \in H_p} \texttt{presence}(z_h) + w_7 \sum_{h \in H_n} \texttt{presence}(z_h) \\
& + w_8 \sum_{i \in I_E \cup I_0} \sum_{h \in H_i \setminus \{s_i\}} \texttt{presence}(y_{ih}) + w_9 \sum_{i \in I_E \cup I_0} b_i^{II} + w_{10} \sum_{i \in I_E \cup I_0} b_i^I + w_{11} \sum_{i \in I} d_i. \qquad (1)
\end{aligned}
$$

Constraints (2) enforce each flight $i \in I$ to be executed by a single helicopter $h \in H_i$, if any. Alternatively, with the predicate `alternative`, these constraints ensure that if flight $i \in I$ is performed (i.e, interval variable x_i is present in the solution) then a single helicopter h out of those

in set H_i must be selected (i.e., the respective interval variable y_{ih} is also present in the solution). In this case, these constraints also ensure that the start and end times of interval variables x_i and y_{ih} are the same; in contrast, all the other interval variables $y_{ih'}$, $h' \in H_i, h' \neq h$, are absent in the solution. Moreover, these constraints ensure that if flight $i \in I$ is not performed (i.e, interval variable x_i is absent in the solution) then all the interval variables y_{ih}, $h \in H_i$, are also absent in the solution.

$$\texttt{alternative}(x_i, \{y_{ih}\}_{h \in H_i}), \qquad i \in I. \quad (2)$$

Constraints (3) use the predicate `span` over an interval variable (helicopter z_h, $h \in H$) and a set of interval variables (flights $\{y_{ih}\}_{i \in I_h}$). They state that the interval of helicopter z_h starts together with the first present interval from flights $\{y_{ih}\}_{i \in I_h}$ and ends together with the last one. Moreover, these constraints ensure that the interval of helicopter z_h is absent if and only if all intervals in flights $\{y_{ih}\}_{i \in I_h}$ are absent in the solution.

$$\texttt{span}(z_h, \{y_{ih}\}_{i \in I_h}), \qquad h \in H. \quad (3)$$

Recall that the possible values of an interval sequence variable are all the permutations of its set of interval variables; however, the interval of this set of variables may overlap with each other. Constraints (4a) use the predicate `noOverlap` over a set of interval variables ($\{y_{ih}\}_{i \in I_h}$), which is represented here by interval sequence variable π_h, $h \in H$, to ensure that all the present intervals in the set are pairwise non-overlapping. Alternatively, it means that whenever each pair of interval variables concerning flights y_{ih} and $y_{i'h}$, $i, i' \in I_h$, $i \neq i'$, are present concerning helicopter $h \in H$, y_{ih} is constrained to end before the start of $y_{i'h}$ or $y_{i'h}$ is constrained to end before the start of y_{ih}. The second and third arguments in predicate `noOverlap` are responsible for modeling the minimal time interval *tat* on the ground at the airport between two flights in a row of the same helicopter. In particular, these arguments ensure a matrix distance $[tat]_{|I_h| \times |I_h|}$ (i.e., all the entries of a predefined matrix of size $|I_h| \times |I_h|$ have the value of *tat*) between each pair of flights $i, i' \in I_h$ in the interval sequence variable π_h, $h \in H$. In other words, the end and start times of two consecutive flights of this helicopter h, beyond non-overlapping, must have a time interval between them as defined by parameter *tat* (e.g., 45 min). Constraints (4b) enforce that whenever an interval variable y_{ih}, $h \in H$, $i \in I_h \cap I_E$, is present in the solution, it must be ordered last in the interval sequence variable π_h in order to guarantee that the helicopter h is blocked to perform other flights after the completion of this entourage flight.

$$\texttt{noOverlap}(\pi_h, [tat]_{|I_h| \times |I_h|}, 0), \qquad h \in H, \quad (4a)$$

$$\texttt{last}(\pi_h, y_{ih}), \qquad h \in H, i \in I_h \cap I_E. \quad (4b)$$

Constraints (5) are logic constraints that model the precedence constraint of the runaway of the onshore airport. They ensure that if two flights $i, j \in I$, $i < j$, are present in the solution then there is a minimum time interval t^a between their departure/start times. Notice that the predicate `start` returns the start time of an interval variable.

$$\texttt{presence}(x_i) + \texttt{presence}(x_j) = 2 \rightarrow |\texttt{start}(x_i) - \texttt{start}(x_j)| \geq t^a, \qquad i, j \in I, i < j. \quad (5)$$

Constraints (6a) and (6b) are precedence constraints that impose the departure/start time of a flight precedes the departure/start of another flight. In particular, constraints (6a) guarantee that the sum of the departure/start time of a transferred flight x_i, $i \in (I_1 \cup I_2) \cap I_p$, plus the value of parameter t_i^u must be lower than or equal to the value of the departure/start time of a table flight x_j, $j \in I_0 \cap I_p$ whenever both flights are present in the solution; notice that these two flights have the same maritime unit $p \in P$ as a destination. Constraints (6b) guarantee that the sum of the departure/start time of a non-entourage flight x_i, $i \in (I_0 \cup I_1 \cup I_2) \cap I_p$, plus the value of parameter t_i^u must be lower than or equal to the value of the departure/start time of an entourage flight x_j, $j \in I_E \cap I_p$, whenever both flights have the same maritime unit $p \in P$ as a destination and are both present in the solution. These constraints block maritime unit $p \in P$ from receiving other flights after the completion of the entourage flight $j \in I_E \cap I_p$.

$$\texttt{startBeforeStart}(x_i, x_j, t_i^u), \qquad p \in P, i \in (I_1 \cup I_2) \cap I_p, j \in I_0 \cap I_p, \quad (6a)$$
$$\texttt{startBeforeStart}(x_i, x_j, t_i^u), \qquad p \in P, i \in (I_0 \cup I_1 \cup I_2) \cap I_p, j \in I_E \cap I_p. \quad (6b)$$

Constraints (7) are logic constraints that model the precedence constraint of the heliport of each maritime unit $p \in P$. They ensure that if two flights $i, j \in I_p \backslash I_E$, $i \neq j$, are present in the solution and flight i departures/starts before flight j, then there is a minimum time interval of t_i^u between their departure/start times.

$$\begin{aligned} &\texttt{presence}(x_i) + \texttt{presence}(x_j) + (\texttt{start}(x_i) < \texttt{start}(x_j)) = 3 \qquad (7) \\ &\quad \rightarrow \texttt{start}(x_i) + t_i^u \leq \texttt{start}(x_j), \qquad p \in P, i, j \in I_p \backslash I_E, i \neq j. \end{aligned}$$

Constraints (8a) are logic constraints that ensure the delay d_i of flight $i \in I$ to be the difference between its departure/start time minus its release time if the interval of flight x_i is present in the solution. Constraints (8b) and (8c), with their respective penalties in the objective function, represent a piecewise linear function that models the behavior of the two types of delays of a flight $i \in I_E \cup I_0$. Notice that constraints (8b) allow each flight $i \in I_E \cup I_0$ to not be delayed ($b_i^I + b_i^{II} = 0$), to have a delay of type I ($b_i^I = 1$ and $b_i^{II} = 0$), or to have a delay of type II ($b_i^I = 0$ and $b_i^{II} = 1$).

$$\texttt{presence}(x_i) = 1 \rightarrow d_i = \texttt{start}(x_i) - r_i, \qquad i \in I, \quad (8a)$$
$$b_i^I + b_i^{II} \leq 1, \qquad i \in I_E \cup I_0, \quad (8b)$$
$$d_i \leq d_{max}^I b_i^I + d_{max}^{II} b_i^{II}, \qquad i \in I_E \cup I_0. \quad (8c)$$

Constraints (9) define the domain of the decision variables. Notice that for an interval variable: the domain $[a, b)$ means that a is the earliest start time and b is the latest end time; the predicate `optional` means that such interval variable may be present or absent in the solution; and, predicate `size` defines the duration of the interval variable. In addition, we observe that parameter σ_h in constraint (9d) is an integer array of size $|I_h|$ with all the elements in I_h, which is required for the

distance matrix $[tat]_{|I_h|\times|I_h|}$ used in constraints (4a).

$$\texttt{interval}\ x_i \subseteq [r_i, tw^b),\ \texttt{optional},\ \texttt{size} = t_i, \qquad i \in I, \quad (9a)$$

$$\texttt{interval}\ y_{ih} \subseteq [r_i, tw^b),\ \texttt{optional},\ \texttt{size} = t_i, \qquad i \in I, h \in H_i, \quad (9b)$$

$$\texttt{interval}\ z_h \subseteq [tw^a, tw^b),\ \texttt{optional}, \qquad h \in H, \quad (9c)$$

$$\texttt{interval sequence}(\pi_h, \{y_{ih}\}_{i\in I_h}, \sigma_h), \qquad h \in H, \quad (9d)$$

$$b_i^I, b_i^{II} \in \{0, 1\}, \qquad i \in I_E \cup I_0, \quad (9e)$$

$$d_i \in \mathbb{Z}_+, \qquad i \in I. \quad (9f)$$

Having defined all the needed elements, we state the model (10), which is a novel CP model for the ARP-PD based on interval variables and interval sequence variables.

$$\textbf{Min} \quad (1), \quad (10a)$$

$$\textbf{s.t.} \quad (2), (3), (4), (5), (6), (7), (8), (9). \quad (10b)$$

4. Iterated Local Search

Due to the complex nature of our problem as well as its multiple, different decision layers, we propose a hybrid method based on the iterated local search (ILS) metaheuristic (Lourenço et al., 2019). The solution method tackles the decisions of the problem individually via different improvement operators. The method starts by attempting to build an initial solution from the instance logs, which is improved by applying a local search algorithm. The heuristic then enters its main loop, alternating between the application of a perturbation algorithm and the local search algorithm. Throughout the execution of these steps, the method builds a set of routes, which is fed to a column generation (CG) formulation of the problem (together with the best solution found). This formulation is used as a one-shot post-optimization step at the end of the execution of the heuristic. An outline of the general structure of the method is shown in Algorithm 1.

Algorithm 1: Iterated local search-based heuristic algorithm for the ARP-PD.

```
1  begin
2      s^best = ∅, f(∅) = ∞
3      Generate an initial solution s^0 (from company logs)
4      s^best = LocalSearch(s^0)
5      while stopping criteria are not met do
6          s' = Perturbation(s^best)
7          s' = LocalSearch(s')
8          if f(s') < f(s^best) then
9              s^best = s'
10         Increase perturbation level, if required
11     Solve CG model using solution pool R and s^best
```

The heuristic includes in its main loop a mechanism to detect stagnation in the search process (line 10). Every time the heuristic reaches a predefined number of iterations without improvement

of the incumbent solution, its perturbation level of the method is increased aiming at helping the method escape from the current local optimal solution. Details of this mechanism as well as all the heuristic components are presented below. Furthermore, the CG formulation of the problem is presented in Appendix.

4.1. Construction heuristic

The construction heuristic from our method uses the instance logs to try to obtain an initial feasible solution to the problem. These files represent the initial desired routing plan by the company and can contain infeasibilities. In these logs, each flight has a preassigned helicopter and a target departure time from the airport. From this information, the heuristic builds a solution by considering the assignments and sequence in the logs, removing any flight that causes infeasibilities in terms of the problem constraints. The routes of each helicopter in this initial solution, if any, are stored in the pool of routes R of the algorithm.

4.2. Local search heuristic

Our local search heuristic tries to improve its incumbent solution by gradually optimizing its sequences of flights using a set of three local search operators. The heuristic iteratively selects one of the operators at random, which is applied to the incumbent solution. If the operator fails to improve the solution, we remove it from the set. Otherwise, if the operator improves the solution we restore the set to its initial configuration, containing all three local search operators. The local search heuristic stops when the set is empty, i.e., when none of the operators can improve the incumbent solution.

The set of local search operators is composed as follows:

- **Insert flights**: this operator tries to insert flights not yet routed into the current sequence. The operator applies the best-improvement strategy by testing the insertion of all the flights that have not yet been routed and trying to insert them into every possible position of the airport sequence using all the helicopters that can execute the flight. From all the feasible insertions, if any, the one with the maximum total cost reduction is executed and the operator is restarted until no insertion is possible.

- **Change helicopter**: the operator attempts to change the helicopter serving each scheduled flight. Using a best-improvement strategy, the operator starts by randomly selecting one of the flights and testing all the alternative helicopters that can carry it out. Note that a change in the helicopter does not result in any change in the airport sequence but in the helicopter sequence. From all the feasible changes for the flight, the operator keeps the one with the largest total cost reduction. The search continues with another randomly chosen flight until no flights are left to test.

- **Or-opt-1**: using a first-improvement strategy, this operator tries to transfer each flight (starting from the first in the airport sequence) from its current position to all the other feasible positions in the sequence. If an improvement transfer is found, the search is restarted until no transfers result in an improvement of the total cost.

4.3. Perturbation algorithm

The perturbation mechanism of our method has a similar structure to our local search algorithm. In particular, the perturbation mechanism uses a set of two operators to gradually change the solution. In each iteration, one operator is randomly chosen to change the incumbent solution. If an operator fails to change the solution, then it is removed from the set and the perturbation phase continues with the remaining operator. On the other hand, if one of the operators changes the solution, the set is re-established to its original form. The algorithm stops when the set is empty or when the number of successful perturbations reaches the predefined level `n_pert`.

The set of perturbation operators contains the following operators:

- **Remove flights**: contrary to the local search component, this operator tries to remove routed flights. The operator removes only one flight every time it is called, choosing it randomly. Notice that the removal of any flight is always feasible.
- **Change helicopter**: this operator attempts to change the helicopter of every scheduled flight. By randomly choosing a flight, the operator iterates randomly over the helicopter that can operate it and checks the feasibility of altering the solution by using the new flight. The operator stops as soon as a feasible change is detected.

4.3.1. Increased perturbation phase

To avoid stalling in local optimal solutions, our heuristic includes a mechanism to increase the strength of the perturbation. This mechanism tries to identify stagnation in the search process and then increase the predefined perturbation level `n_pert`. In particular, when the number of iterations (of the outer loop) without improvement (`iter_no_imp`) is a multiple of `iter_to_incr`, the perturbation level `n_pert` is increased by `add_to_pert` $*$ `iter_no_imp`/`iter_to_incr` during `n_incr_pert` iterations. Note that we need `n_incr_pert` $<$ `iter_no_imp`.

5. Computational experiments

In this section, we report the experiments performed to evaluate the computational performance of the CP model and hybrid ILS. We refer to the CP model (10) of Section 3 as CP-Model. We refer to the ILS algorithm of Section 4 as ILS-CG (resp. ILS) whether considering (resp. not considering) the column generation formulation of the Appendix as a one-shot post-optimization step. We considered as benchmark the exact and heuristic approaches proposed by Vieira et al. (2021), which are the current state-of-the-art solution approaches for the ARP-PD. Concerning their approaches, we denote their exact network flow model as MILP-Model and heuristic as Constructive Heuristic. We used IBM CPLEX Optimization Studio v.22.1 as MILP and CP solver. The proposed and benchmark approaches were coded in C++, unless to the results for Constructive Heuristic which were taken from table results of Vieira et al. (2021). All experiments were carried out on a PC with Intel Xeon E5-2680v2 (2.8 GHz), using a single thread, 16 GB RAM, under a CentOS Linux 7.2.1511 Operating System. Each run of the approaches was limited to 3,600 seconds unless stated otherwise. Specifically for the ILS-CG, the time limit is divided between two parts, that is, one half for the ILS and the other half for the post-optimization step with the column generation formulation. We use the letters "tl" to indicate when this time limit was reached for a given problem instance.

Table 1: Sets of benchmark problem instances.

Sets	Number of instances	Number of flights ($\overline{I}$)	Number of helicopters ($\overline{H}$)	Number of maritime units ($\overline{P}$)
#A	20 instances	8 to 45	3 to 13	5 to 27
#B	72 instances	37, 38, or 45	11	17 to 27

We report information on the two sets of benchmark problem instances used in the experiments in Table 1, which makes 92 problem instances in total. We consider the same problem instances of Vieira et al. (2021) – we refer the reader to this work for a detailed description of the instances. The set of instances #A consists of twenty real-life-based instances provided by a Brazilian oil company. Their main information are described in Table 2. The set of instances #B is based on variations of the three largest-sized instances from set of instances #A (I37C, I38C, and I45C), that is, scenarios 1 to 8 of Vieira et al. (2021).

Table 2: Information of the instances of set #A.

Instances	$\overline{I}$	$\lvert H_n\rvert$	$\lvert H_p\rvert$	$\lvert H_s\rvert$	$\overline{P}$	$\lvert I_0\rvert$	$\lvert I_1\rvert$	$\lvert I_2\rvert$	$\lvert I_E\rvert$
I8A	8	2	1	0	6	6	1	1	0
I9A	9	2	1	0	5	7	1	1	0
I10A	10	2	1	0	7	8	1	1	0
I11A	11	2	3	1	10	7	3	0	1
I12A	12	3	0	0	9	11	1	0	0
I13A	13	2	1	1	10	7	5	1	0
I14A	14	4	2	1	11	7	4	2	1
I15B	15	3	2	2	11	11	2	1	1
I18B	18	6	2	0	15	11	5	0	2
I20B	20	4	3	1	12	14	6	0	0
I22B	22	4	1	1	13	17	5	0	0
I25B	25	12	0	0	20	20	5	0	0
I27B	27	10	3	0	22	21	6	0	0
I38B	28	6	2	0	19	13	13	2	0
I30B	30	7	2	2	20	12	16	1	1
I33C	33	5	2	1	21	12	18	3	0
I35C	35	12	0	0	27	22	10	2	1
I37C	37	11	0	0	17	30	7	0	0
I38C	38	11	0	0	24	15	20	3	0
I45C	45	11	0	0	27	22	20	3	0

5.1. Results for the set of instances #A

We report the results considering the instance set #A in Table 3. For each of the four approaches, we report the value of the objective function (ofv), optimality gap in percentage (gap[%]), and computing time in seconds (time[s]). As for the MILP-Model and CP-Model, we also report the value of the lower bound (lb). For each instance, we set the gap as $100 \cdot (\text{ofv} - \overline{lb})/\text{ofv}$, where $\overline{lb}$ is the best lower bound found among the two exact approaches. For each instance, the approach(es) with the best value of the objective function are highlighted in bold, except when all

	MILP-Model				Constructive Heuristic			CP-Model				ILS-CG		
Instance	lb	ofv	gap[%]	time[s]	ofv	gap[%]	time[s]	lb	ofv	gap[%]	time[s]	ofv	gap[%]	time[s]
I8A	*42,13*	42,13	0,00	0,06	42,13	0,00	0,02	*42,13*	42,13	0,00	0,03	42,13	0,00	6,16
I9A	*67,03*	67,03	0,00	0,15	67,03	0,00	0,07	*67,03*	67,03	0,00	0,27	67,03	0,00	14,60
I10A	*66,64*	**66,64**	0,00	1,15	66,65	0,01	0,05	*66,64*	**66,64**	0,00	0,32	**66,64**	0,00	22,54
I11A	*91,03*	**91,03**	0,00	5,01	102,40	11,11	0,08	*91,03*	**91,03**	0,00	0,09	**91,03**	0,00	24,00
I12A	*71,19*	**71,19**	0,00	2,56	72,18	1,38	0,08	*71,19*	**71,19**	0,00	0,13	**71,19**	0,00	33,65
I13A	*68,13*	**68,13**	0,00	289,97	88,63	23,14	0,70	*68,13*	**68,13**	0,00	0,70	**68,13**	0,00	66,32
I14A	*83,68*	**83,68**	0,00	1968,53	83,98	0,35	0,15	*83,68*	**83,68**	0,00	1,24	**83,68**	0,00	95,45
I15B	90,43	163,38	11,64	tl	161,48	10,60	0,18	*144,36*	**144,36**	0,00	19,71	**144,36**	0,00	121,01
I18B	123,07	148,18	0,18	tl	148,20	0,19	0,81	*147,92*	**147,92**	0,00	26,06	147,95	0,02	172,20
I20B	68,71	108,33	0,08	tl	130,60	17,12	1,38	*108,24*	**108,24**	0,00	4,97	**108,24**	0,00	322,24
I22B	96,42	142,97	13,33	tl	131,28	5,62	1,70	*123,91*	**123,91**	0,00	13,74	131,27	5,61	389,82
I25B	*99,04*	215,05	53,95	tl	208,67	52,54	3,76	46,73	**206,21**	51,97	tl	206,61	52,07	2712,81
I27B	*127,91*	**186,48**	31,41	tl	207,71	38,42	6,16	55,17	**186,48**	31,41	tl	**186,48**	31,41	3022,62
I28B	*87,43*	594,51	85,29	tl	211,63	58,69	6,27	75,79	**167,06**	47,66	tl	179,72	51,35	3311,21
I30B	*47,23*	397,20	88,11	tl	201,93	76,61	13,79	44,43	**163,92**	71,19	tl	184,73	74,43	tl
I33C	87,76	677,08	86,83	tl	260,70	65,80	11,77	*89,15*	**214,67**	58,47	tl	244,49	63,54	2705,13
I35C	*127,46*	252,86	49,59	tl	250,83	49,18	24,22	55,25	228,04	44,10	tl	**227,08**	43,87	tl
I37C	*107,75*	337,13	68,04	tl	243,77	55,80	28,94	50,93	**237,54**	54,64	tl	239,77	55,06	tl
I38C	*46,64*	1416,58	96,71	tl	214,96	78,30	63,74	44,61	**185,95**	74,92	tl	205,91	77,35	tl
I45C	*58,93*	2396,55	97,54	tl	277,41	78,76	109,63	47,67	251,37	76,56	tl	**249,36**	76,37	tl

Table 3: Results for the set of instances #A.

four approaches reached the same value. We also highlight in italics the approach(es) with best value of the lower bound.

Concerning the results in Table 3, it is worth noting that the number of best solutions found by the MILP-Model is 8 instances, for Constructive Heuristic is 2 instances, for the solver with CP-Model is 18 instances, and for ILS-CG is 12 instances, out of 20 instances. As for the value of the lower bound, the solver with MILP-Model reaches the best value for 15 instances and with CP-Model for 12 instances. The optimality was proven by the solver with MILP-Model for 7 instances and with CP-Model for 11 instances. We highlight that the optimality of solutions for instances I14A, I15B, I18B, I20B, and I22B are proven by the first time in the literature with CP-Model; notice that in our experiments the solver with MILP-Model also proves the optimality for instance I14A. Among the two exact approaches, the CP-Model outperforms the MILP-Model in terms of the number of best solutions and optimal solutions, but not in terms of value of the lower bound. Similarly, among the two heuristic approaches, the ILS-CG clearly outperforms the Constructive Heuristic in number of best solutions, but not in relation to computational time. In summary, our approaches are very competitive in relation to approaches in the literature to address these small, medium and large sized instances of set #A.

5.2. Results for the set of instances #B

We report the results considering the instance set #B in Table 4. Each entry of the table is an average of 24 instances. The objective of these experiments is to evaluate the computational performance of the approaches, with different time limit values (tl=60,300,600,1200,1800,3600 seconds), in relation to the best solution found among the four approaches. For these experiments, given the results of the previous section regarding the Constructive Heuristic (i.e., fast but of lower quality), we chose not to consider it, using the ILS as an alternative heuristic approach. However, we note that the results of our approaches outperformed the results of Constructive Heuristic for the set of instances #B. As for the four approaches, we report the number of variables (var.) and constraints (cons.) (except for the ILS), the value of the objective function (ofv), and the deviation with respect to the best solution found among the four approaches in percentage (dev.[%]). We set the deviation as $100 \cdot (\text{ofv} - \overline{\text{ofv}})/\overline{\text{ofv}}$, where $\overline{\text{ofv}}$ is the value of the best solution found among the four approaches (concerning the 24 instances of the entry). We note that the solver with CP-Model finds high-quality feasible solutions very quickly and that the column generation formulation benefits the ILS-CG over the ILS as the time limit increases. In average terms, the best results are obtained first by the solver with CP-Model, followed by the ILS-CG and ILS, and lastly the solver with MILP-Model. Finally, the experiments show that our approaches outperform literature approaches in several types of problem instances for the ARP-PD. We understand that the CP-Model obtains the best results although dependent on a general-purpose CP solver, while ILS-CG or ILS can be a competitive approach in scenarios without such a resource.

6. Conclusions

We addressed a challenging practical problem of short-term flight rescheduling in the oil and gas industry, with characteristics such as diverse flight types, helicopter compatibility, and operational constraints, motivated by a Brazilian oil and gas practical application. For the problem, we proposed an exact approach based on a constraint programming model and a heuristic approach

		I37C-based instances				I38C-based instances				I45C-based instances			
Approach	tl[s]	var.	cons.	ofv	dev.[%]	var.	cons.	ofv	dev.[%]	var.	cons.	ofv	dev.[%]
MILP-Model	60	490,284	932,872	1,550.82	568.60	514,542	980,063	4,091.88	2,047.91	707,424.00	1,354,142	4,871.44	1,859.90
	300	490,284	932,872	731.74	212.06	514,542	980,063	2,926.02	1,434.70	707,424.00	1,354,142	3,992.74	1,504.23
	600	490,284	932,872	523.09	121.70	514,542	980,063	1,827.31	864.20	707,424.00	1,354,142	3,295.79	1,218.90
	1200	490,284	932,872	385.88	65.27	514,542	980,063	1,633.62	760.59	707,424.00	1,354,142	2,463.71	885.67
	1800	490,284	932,872	376.48	62.05	514,542	980,063	1,483.95	685.62	707,424.00	1,354,142	2,119.38	751.83
	3600	490,284	932,872	340.25	45.92	514,542	980,063	1,321.54	593.22	707,424.00	1,354,142	1,746.32	598.97
CP-Model	60	11,791	22,650	246.90	5.55	11,299	22,690	205.97	7.51	13,551.00	32,225	266.86	7.01
	300	11,791	22,650	240.38	2.67	11,299	22,690	194.48	1.42	13,551.00	32,225	260.49	4.41
	600	11,791	22,650	238.58	1.91	11,299	22,690	193.97	1.17	13,551.00	32,225	257.49	3.21
	1200	11,791	22,650	237.29	1.35	11,299	22,690	193.21	0.77	13,551.00	32,225	255.43	2.47
	1800	11,791	22,650	234.97	0.38	11,299	22,690	192.82	0.59	13,551.00	32,225	254.72	2.19
	3600	11,791	22,650	234.55	0.20	11,299	22,690	191.70	0.00	13,551.00	32,225	250.98	0.63
ILS	60	-	-	247.19	5.67	-	-	229.03	19.35	-	-	284.64	14.41
	300	-	-	243.41	3.97	-	-	219.21	14.40	-	-	269.14	7.97
	600	-	-	243.41	3.97	-	-	214.57	12.02	-	-	266.32	6.83
	1200	-	-	243.41	3.97	-	-	210.88	9.98	-	-	265.74	6.60
	1800	-	-	243.41	3.97	-	-	210.57	9.80	-	-	265.62	6.56
	3600	-	-	243.41	3.97	-	-	210.22	9.59	-	-	265.62	6.56
ILS-CG	60	41,497	396,132	247.71	5.86	-	416,611	231.70	20.81	-	-	288.88	16.05
	300	49,660	396,132	241.24	3.01	-	416,611	218.22	13.92	-	-	269.28	8.01
	600	59,749	396,132	239.81	2.36	-	416,611	212.74	11.13	-	-	264.57	6.12
	1200	74,851	396,132	239.13	2.17	-	416,611	209.81	9.45	-	-	262.06	5.14
	1800	82,099	396,132	237.94	1.68	-	416,611	207.73	8.39	-	-	260.16	4.41
	3600	97,914	396,132	237.94	1.68	-	416,611	207.30	8.07	-	-	260.16	4.41

Table 4: Results for the set of instances #B.

based on a hybrid iterated local search algorithm. The proposed approaches demonstrate promising results in achieving full flight recovery and minimizing delays within acceptable computational times. The results of the experiments showed that our approaches outperformed approaches found in the literature in terms of the number of proven optimal solutions and quality of feasible solutions, within the acceptable running times for the application.

Future studies may explore enhanced solution methods, including branch-and-cut approaches and meta-heuristic techniques, to address larger instances and improve solution quality. We emphasize that obtaining lower bounds for this problem has proven difficult, which could be overcome by developing valid inequalities or reduction techniques for the problem. Additionally, considering uncertainties in problem parameters and extending models to accommodate various real-world scenarios present interesting opportunities for advancing research in this field.

Disclosure of interest

The authors report there are no competing interests to declare.

Acknowledgements

The authors thank the financial support of the São Paulo Research Foundation (FAPESP-Brazil) [grant number 2022/05803-3] and the National Council for Scientific and Technological Development (CNPq-Brazil) [grants numbers 408722/2023-1 and 405702/2021-3]. The research was carried out using the computational resources of the Center for Mathematical Sciences Applied to Industry (CeMEAI), funded by FAPESP-Brazil [grant number 2013/07375-0].

Appendix. Column generation formulation for the ARP-PD

This section presents the column generation formulation for the problem. The formulation takes a set of routes R as input and returns the best possible schedule resulting from synchronizing those routes at the airport and maritime units.

To introduce the formulation, consider the following additional notation. Let R_h be the set of routes of helicopter h. For a given route r and helicopter h, let a^i_{rh} be a binary indicator taking value one if and only if flight i is carried our by the helicopter. Similarly, b^{ij}_{rh} indicates whether or not flights i and j are performed consecutively by the aircraft. Finally, I_r is the set of flights of the route. Note that p_h represents the type of helicopter associated to h. By construction, the routes are feasible regarding the order of the flights and compatibility between the helicopter type and the flights on the route. Consider the following decision variables:

- λ_{rh} binary variable which equals 1 if route r of helicopter h is selected for the solution, and 0 otherwise;
- Z_{ij} binary variable which equals 1 if flight i precedes flight j in the airport sequence, and 0 otherwise;
- DT_i continuous variable representing the departure time of flight i from the airport;
- AT_i continuous variable representing the arrival time of flight i at the airport.

Using this notation in addition to the one presented in Sections 2 and 3, we state a CG formulation for the ARP-PD as follows:

$$\textbf{Min}\ w_1|I_E| + w_2|I_2| + w_3|I_1| + w_4|I_0| + \sum_{h\in H}\sum_{r\in R_h} c_{rh}\lambda_{rh} + w_9 \sum_{i\in I_E\cup I_0} b_i^{II} + w_{10} \sum_{i\in I_E\cup I_0} b_i^{I} + w_{11}\sum_{i\in I} d_i, \quad (11)$$

$$\textbf{s.t.}\ \sum_{r\in R_h} \lambda_{rh} \le 1, \qquad h\in H, \quad (12)$$

$$\sum_{h\in H_i}\sum_{r\in R_h} a_{rh}^i \lambda_{rh} \le 1, \qquad i\in I, \quad (13)$$

$$DT_j \ge AT_i + tat \sum_{r\in R_h} b_{rh}^{ij}\lambda_{rh} - M\left(1 - \sum_{r\in R_h} b_{rh}^{ij}\lambda_{rh}\right), \qquad i,j\in I, i\ne j,\ h\in H_i\cap H_j, \quad (14)$$

$$AT_i \ge DT_i + t_i \sum_{h\in H_i}\sum_{r\in R_h} a_{rh}^i\lambda_{rh}, \qquad i\in I, \quad (15)$$

$$r_i \sum_{h\in H_i}\sum_{r\in R_h} a_{rh}^i\lambda_{rh} \le DT_i, \qquad i\in I, \quad (16)$$

$$DT_i \le r_i \sum_{h\in H_i}\sum_{r\in R_h} a_{rh}^i\lambda_{rh} + d_i, \qquad i\in I, \quad (17)$$

$$AT_i \le tw^b \sum_{h\in H_i}\sum_{r\in R_h} a_{rh}^i\lambda_{rh}, \qquad i\in I, \quad (18)$$

$$Z_{ij} + Z_{ji} \le 1, \qquad i,j\in I, i<j, \quad (19)$$

$$Z_{ij} + Z_{ji} \ge \sum_{h\in H_j}\sum_{r\in R_h} a_{rh}^j\lambda_{rh} + \sum_{h\in H_i}\sum_{r\in R_h} a_{rh}^i\lambda_{rh} - 1, \qquad i,j\in I, i<j, \quad (20)$$

$$DT_j - DT_i \le M\left(Z_{ij} + 2 - \sum_{h\in H_j}\sum_{r\in R_h} a_{rh}^j\lambda_{rh} - \sum_{h\in H_i}\sum_{r\in R_h} a_{rh}^i\lambda_{rh}\right), \qquad i,j\in I, i\ne j, \quad (21)$$

$$DT_j - DT_i \ge t^a Z_{ij} - M(1-Z_{ij}), \qquad i,j\in I, i\ne j, \quad (22)$$

$$DT_j - DT_i \ge t_i^u Z_{ij} - M(1-Z_{ij}), \qquad p\in P,\ i,j\in I_p, i\ne j, \quad (23)$$

$$d_i \le d_{max}^{II} \sum_{h\in H_i}\sum_{r\in R_h} a_{rh}^i\lambda_{rh}, \qquad i\in I_E\cup I_0, \quad (24)$$

$$d_i \le d_{max}^{I} b_i^I + d_{max}^{II} b_i^{II}, \qquad i\in I_E\cup I_0, \quad (25)$$

$$b_i^I + b_i^{II} \le 1, \qquad i\in I_E\cup I_0, \quad (26)$$

$$\lambda_{rh}\in\{0,1\}, \qquad h\in H,\ r\in R_h, \quad (27)$$

$$Z_{ij}\in\{0,1\}, \qquad i,j\in I, i\ne j, \quad (28)$$

$$b_i^I, b_i^{II}\in\{0,1\}, \qquad i\in I_E\cup I_0, \quad (29)$$

$$DT_i, AT_i, d_i \ge 0, \qquad i\in I, \quad (30)$$

in which the cost and prizes for each helicopter $h \in H$ in route $r \in R_h$ are computed as:

$$c_{rh} = -\sum_{i \in I_r \cap I_E} w_1 - \sum_{i \in I_r \cap I_2} w_2 - \sum_{i \in I_r \cap I_1} w_3 - \sum_{i \in I_r \cap I_0} w_4 + w_{p_h} + \sum_{\substack{i \in I_r, \\ h \neq s_i}} w_8. \tag{31}$$

The objective function (11) consists of minimizing the total cost, encompassing the eleven terms defined in Section 3 and their corresponding penalties. Notice that the objective function starts including all the penalties for not attending any flight (first four terms) and we favor selecting routes by giving a price for attending flights in Equation (31).

Constraints (12) guarantee that at most one route for each helicopter is selected in the solution. Constraints (13) impose that each flight is scheduled at most once (given the existence of the prizes mentioned above). Constraints (14) synchronize in the airport the departure of consecutive flights carried out by the same helicopter. Constraints (15) compute the arrival time at the airport of each flight. Constraints (16) allow the departure of flights only after their scheduled time, constraints (17) compute the delay with respect to this departure time while the closing time of the airport is ensured by constraints (18). Constraints (19) define that at most one precedence between flights i and j exists whereas constraints (20) enforce one of these precedence if both flights are scheduled. The synchronization at the airport for scheduled flights is guaranteed by constraints (21), while constraints (22) and (23) impose the minimum time between flights at the airport and maritime units, respectively. Constraints (24) enforce the maximum allowed delay of flights while the delay type is captured by constraints (25). Constraints (26) state that only one type of delay is possible. The domain of the decision variables is stated in constraints (27) to (30).

References

Clausen, J., Larsen, A., Larsen, J., & Rezanova, N. J. (2010). Disruption management in the airline industry—concepts, models and methods. *Computers & Operations Research*, *37*, 809–821. doi:https://doi.org/10.1016/j.cor.2009.03.027. Disruption Management.

De La Vega, J., Santana, M., Pureza, V., Morabito, R., Bastos, Y., & Ribas, P. C. (2022a). Model-based solution approach for a short-term flight rescheduling problem in aerial passenger transportation to maritime units. *International Transactions in Operational Research*, *29*, 3400–3434. doi:10.1111/itor.13079.

De La Vega, J., Vieira, T., Santana, M., Pureza, V., Morabito, R., Tavares, R., Bastos, Y., & Ribas, P. C. (2022b). Helicopter recovery in an oil and gas industry: Model and solution approaches. *EURO Journal on Transportation and Logistics*, *11*, 100084. doi:10.1016/j.ejtl.2022.100084.

Filar, J. A., Manyem, P., & White, K. (2001). How airlines and airports recover from schedule perturbations: A survey. *Annals of Operations Research*, *108*, 315–333. doi:https://doi.org/10.1023/A:1016079600083.

Hassan, L., Santos, B., & Vink, J. (2021). Airline disruption management: A literature review and practical challenges. *Computers & Operations Research*, *127*, 105137. doi:https://doi.org/10.1016/j.cor.2020.105137.

Kohl, N., Larsen, A., Larsen, J., Ross, A., & Tiourine, S. (2007). Airline disruption management—perspectives, experiences and outlook. *Journal of Air Transport Management*, *13*, 149–162. doi:https://doi.org/10.1016/j.jairtraman.2007.01.001.

Lourenço, H. R., Martin, O. C., & Stützle, T. (2019). Iterated local search: Framework and applications. In M. Gendreau, & J.-Y. Potvin (Eds.), *Handbook of metaheuristics* (pp. 129–168). Springer. doi:10.1007/978-3-319-91086-4_5.

Santana, M., De La Vega, J., Morabito, R., & Pureza, V. (2023). The aircraft recovery problem: A systematic literature review. *EURO Journal on Transportation and Logistics*, *12*, 100117. doi:10.1016/j.ejtl.2023.100117.

Teodorović, D., & Guberinić, S. (1984). Optimal dispatching strategy on an airline network after a schedule perturbation. *European Journal of Operational Research*, *15*, 178–182. doi:https://doi.org/10.1016/0377-2217(84)90207-8.

Van Hentenryck, P. (1999). *The OPL Optimization Programming Language*. Cambridge, MA, USA: MIT Press.

Vieira, T., De La Vega, J., Tavares, R., Munari, P., Morabito, R., Bastos, Y., & Ribas, P. C. (2021). Exact and heuristic approaches to reschedule helicopter flights for personnel transportation in the oil industry. *Transportation Research Part E: Logistics and Transportation Review*, *151*, 102322. doi:10.1016/j.tre.2021.102322.