https://doi.org/10.1016/j.cie.2018.05.002

A model with a solution algorithm for the operational aircraft maintenance routing problem

Abdelrahman E.E. Eltoukhy[a], Felix T.S. Chan[a], S.H. Chung[a], and B. Niu[b, *]

[a]*Department of Industrial and Systems Engineering, The Hong Kong Polytechnic University, Hung Hum, Hong Kong*

[b]*College of Management, Shenzhen University, Shenzhen, China*

*Corresponding Author

elsayed.abdelrahman@connect.polyu.hk

f.chan@polyu.edu.hk

mfnick@polyu.edu.hk

drniuben@gmail.com

wxp@dlut.edu.cn

Abstract

The operational aircraft maintenance routing problem (OAMRP) determines the route for each individual aircraft while incorporating operational maintenance considerations. This problem is significant for airlines as it determines the routes to be flown in real life. Most studies incorporate particular operational maintenance requirements, like restrictions on the total number of days between two successive maintenance checks, while neglecting other considerations. Such neglected considerations include restrictions on the total cumulative flying time, restrictions on the total number of takeoffs, the workforce capacity and the working hours of the maintenance stations. This can result in the generation of routes that are not feasible for implementation in reality. In this paper, we study OAMRP, with two objectives. First, to propose a model that considers all operational maintenance requirements, and for this purpose, we formulate a mixed integer linear programming (MILP) model by modifying the connection network. The proposed model is solved using commercial software for small size problems. Second, to develop a solution algorithm that solves the model efficiently and quickly while tackling medium and large-scale problems. The performance of the proposed solution algorithm is assessed based on real data obtained from EgyptAir. The results demonstrate high quality solutions and significant savings in computational time. The experiments were extended for two reasons. First, to compare the performance of the proposed solution algorithm with existing solution methods. Second, to test the effect of incorporating the operational requirements on profit. The results show that the proposed algorithm outperforms existing methods, like compressed annealing (CA), in producing better solution quality in much shorter computational time. In addition, the results reveal that considering the maintenance workforce capacity improves the profitability of the airline. Such outcomes provide evidence that the proposed model and solution method have great potential for solving the actual OAMRP.

Keywords: Air transportation, Aircraft routing problem, Airline operations, Integer programming.

1. Introduction

In the last decade, the development of the aviation industry has shown radical economic growth. Similarly, passenger demand is currently blooming and showed an increase of 5.2% from 2012 to 2013 (ICAO, 2014), and was predicted to grow by 31% from 2012 to 2017 (IATA, 2014). Despite this pleasing situation for airlines, they have a great challenge in assigning more flights to their aircraft in order to cope with the demand growth, while meeting the operational regulations. Managing any increased number of flights while keeping the aircraft well maintained is a difficult operation for the airline industry. For example, in 2010, 65000 U.S. airline flights could not take off because the aircraft did not receive proper maintenance, resulting in a $28.2 million penalty cost against 25 U.S. airlines (Stoller, 2010). This investigation also

showed the importance of maintenance since it was a cause, factor, or finding in 18 accidents, involving 43 deaths and 60 injuries. In this regard, the aircraft maintenance routing problem (AMRP), which is the main focus of this study, is very significant to airlines in that it generates maintenance feasible routes for each aircraft flown.

AMRP is one of the most studied problems in the aviation literature with two main focuses: tactical and operational. Regarding the tactical side, it aims to generate specific rotations for each aircraft in the fleet, while neglecting many of the operational maintenance constraints. The generated rotations are repeated by each aircraft in the fleet. Using a single rotation for each aircraft is not applicable due to not considering operational maintenance constraints. Thus, AMRP is studied with more operational focus, which aims to specify maintenance feasible routes for each aircraft in the fleet. A route is maintenance feasible when it satisfies the operational maintenance requirements, which are the restrictions on the total cumulative flying time, number of maintenance operations every four days, restrictions on the total number of take-offs, the working hours and the workforce capacity of each maintenance station.

In this paper, we focus on the operational side of AMRP, and our aim is twofold. Firstly, to develop an OAMRP model that considers all the operational maintenance constraints in a single model. For this purpose, a new MILP is proposed for OAMRP. Secondly, to propose an effective solution algorithm that can solve the OAMRP while tackling medium and large-scale instances that cannot be solved using CPLEX. The performance of the proposed algorithm is validated with respect to exact solutions obtained from CPLEX for small size problems, whereas the best upper bound is used to assess the performance while solving medium and large-scale problems. To demonstrate the importance of the proposed algorithm over existing solution methods, we conduct a comparison between our proposed algorithm and one of the existing solution methods, called compressed annealing. In addition, the proposed OAMRP is modified so as to be as similar to those found in the literature. Then, a comparison is conducted between these two versions to capture the implications on profit after considering the maintenance workforce capacity constraints.

The rest of the paper is organized as follows. Section 2 presents the literature review on AMRP and the contribution of this study. In section 3, we describe the OAMRP and the new MILP formulation proposed. The effective solution algorithm is described in section 4. Section 5 covers the comparison between our proposed model and the models proposed in the literature. In section 6, the computational experimental results for real cases are provided. Conclusions to the study are given in section 7.

2. Literature review and contribution

2.1 literature review

In this section, we briefly present and discuss some of the existing models regarding AMRP. In the literature, the focus of AMRP studies can be classified as being either tactical or operational. Regarding the tactical side, the main aim is to generate specific rotations for each aircraft in the fleet, while neglecting many of the operational maintenance constraints. The operational side, on the other hand, aims to specify each aircraft's route beside taking into account the operational maintenance constraints, in order to determine the routes to be flown in real life. There are four main operational maintenance constraints that should be considered while tackling the OAMRP. First, each aircraft should undergo maintenance once every four days. Second, the cumulative flying time between two successive maintenance operations should not exceed the maximum flying time allowed for each aircraft. Third, the total number of take-offs between two successive maintenance operations should not exceed the maximum number of take-offs allowed for each aircraft. The first three constraints are mandated by the Federal Aviation Administration (FAA) (Haouari et al., 2012). Lastly, for each maintenance station, the number of aircraft to be maintained should not exceed the workforce capacity of the maintenance station. Also, the working hours of each maintenance station should be considered.

Focusing on the tactical side of AMRP, (Kabbani & Patty, 1992) formulated AMRP as a set-partitioning model to find feasible routes or lines of flight (LOF) for 3-day AMRP, while the maintenance operations were assumed to be carried out overnight. The use of (LOF) was expanded by (Gopalan & Talluri, 1998)

in order to solve the k-days AMRP. They developed a polynomial time algorithm in order to determine maintenance feasible routes for aircraft for 3-day AMRP. The proposed algorithm was used to solve the static and dynamic formulations of the problem, where the routes were assumed to be fixed or not fixed, respectively. The 2-day AMRP formulation of the problem was shown to be NP-Hard when the routes were not fixed. (Talluri, 1998) developed an effective heuristic to solve the 4-day AMRP, which was shown to be NP-hard even when the routes were fixed. (Clarke et al., 1997) adopted lagrangian relaxation to solve their proposed model that aimed at finding feasible maintenance rotations that could yield the maximum through value. The through value can be defined as the additional profit gained through connecting some specific flights. More recently, (Liang et al., 2011) developed a new rotation-tour time-space network for AMRP and proposed a new integer linear programming (ILP) model according to their network. The proposed model was solved using commercial software. The authors neglected the flight assignment to each individual aircraft and assumed that the maintenance was performed overnight.

Based on the operational side of AMRP, which is called OMARP, (Sriram & Haghani, 2003) presented an ILP model that considered only two among the four maintenance constraints: one maintenance visit every four days and the workforce capacity constraints. The authors adopted Origin-Destination (OD) pairs, similar to LOF. It was assumed that the OD pairs were already determined, and the maintenance operations were performed overnight. The proposed model aimed to assign each aircraft to routes with an objective of minimizing both the maintenance cost and other costs incurred during the re-assignment process. An effective heuristic was proposed, and the model was solved in a reasonable computational time when compared with CPLEX. The authors also extended their model and considered the cumulative flying hours; however, the authors did not attempt to solve it because of its high complexity. It is worth mentioning that the extended model was quite simple compared to our model proposed in this paper. (Sarac et al., 2006) formulated OAMRP as a set-partitioning model that considered only maximum flying hours and workforce capacity as maintenance constraints and neglected the rest. Since the set-partitioning formulation produces an exponential number of feasible routes, the authors adopted column generation as a solution technique. (Haouari et al., 2012) developed a non-linear model for OAMRP, while considering three maintenance constraints simultaneously. These constraints were: one visit every four days, the maximum flying hours, and the maximum number of take-offs. Workforce capacity constraints were also considered in their model, but were relaxed during the computational results. The authors linearized their model by using a reformulation-linearization technique that provided high quality solutions while solving the daily version of OAMRP. (Başdere & Bilge, 2014) developed an ILP model for OAMRP, while considering the maximum flying hours as a maintenance constraint. The proposed model was solved by using both branch-and-bound (B&B) and compressed annealing (CA). The authors reported that CA outperformed B&B for large-scale problems and could find feasible solutions in minutes. Recently, (Eltoukhy et al., 2017a) developed an OAMRP model that considered only the maximum flying hours and the workforce capacity as maintenance constraints. More details regarding the above AMRP models are illustrated in Table 1.

Table 1: Summary of the literature for AMRP.

Author/s	Year	Planning horizon	Network	Model formulation	Objective	Solution procedure	Computational study	
							Data	Airline
(Kabbani & Patty)	1992	3D	-	SP	Min total cost	Developed heuristic	RL	American
(Clarke et al.)	1997	NA	TSN	ATS	Max through value	Lagrangian relaxation and subgradient optimization	RL	U.S.
(Talluri)	1998	4D	-	ET	Finding feasible routes	3-day algorithm and Polynomial time algorithm	NA	-
(Gopalan & Talluri)	1998	3D	CN	ET	Finding feasible routes	Polynomial time algorithm	RL	-
(Sriram & Haghani)	2003	W	CN	ILP	Min maintenance and re-assignment cost	Hybrid of random search and depth first search	G	-

(Sarac et al.)	2006	D	CN	SP	Min number of unused legal flying hours	Branch and price approach	RL	U.S.
(Liang et al.)	2011	D	TSN	NF	Min the connection cost and maintenance cost	CPLEX callable library version 10.0.	RL	U.S. carrier
(Haouari et al.)	2012	D	CN	NLP	Finding feasible routes	CPLEX 12.1	RL	U.S.
(Başdere & Bilge)	2014	W	CN	ILP	Max utilization of Remaining flying time	1st: branch-and-bound 2nd: Compressed annealing heuristic	RL	-

Planning horizon: daily (D), 3-day (3D), 4-day (4D) or weekly (W). Network: connection network (CN) or time-space network (TSN). Model formulation: network flow (NF), integer linear programming (ILP), Euler tour (ET), asymmetric traveling salesman (ATS), set-partitioning (SP) or nonlinear programming (NLP). Used data: real life (RL) or generated (G). non-available (NA).

While many research studies focused only on AMRP, as mentioned above, other research studies focused on integrating AMRP with other airline scheduling phases such as the flight scheduling problem (FSP), the fleet assignment problem (FAP), and the crew scheduling problem (CSP). The papers by (Barnhart et al., 1998), (Zeghal et al., 2011), (Haouari et al., 2011), and (Dong et al., 2016) aimed at integrating FAP and AMRP. On the other hand, the integration between AMRP and CSP appeared in the work by (Klabjan et al., 2002), (Cohn & Barnhart, 2003), (Mercier et al., 2005), (Weide et al., 2010), (Dunbar et al., 2014) and (Díaz-Ramírez et al., 2014). There were some attempts to integrate three phases. For example, FSP, AMRP, and CSP received attention from (Mercier & Soumis, 2007). In addition, (Sherali et al., 2013) and (Gürkan et al., 2016) introduced integration for FSP, FAP, and AMRP. Moreover, some scholars were interested in integration between FAP, AMRP, and CPP, as shown by (Salazar-González, 2014) and (Cacchiani & Salazar-González, 2016). For a recent survey regarding FSP, FAP and CSP, we refer interested readers to the recent review paper by (Eltoukhy et al., 2017b).

The airline industry is often faced by disruptions and unexpected circumstances. This motivates researchers to solve AMRP while considering these disruptions, as appeared in the work by (Ben Ahmed et al., 2017), (Hu et al., 2015), and (Zhang et al., 2015).

From the above work, we can see that there is no AMRP model that considers the three operational maintenance requirements besides the workforce capacity and the working hours of the maintenance stations in a single model. This paper attempts to fill this gap by considering all these issues in a single model and to develop an efficient solution algorithm that can handle medium and large-scale test instances. To our best knowledge, our proposed model is the first one that considers all the mentioned issues in a single model.

2.2 Contribution

The focus of this paper is the operational side of AMRP and the contribution of this work is as follows. Firstly, from the literature, we can see that the set-partitioning or the set covering based formulations are commonly used, in which all the possible feasible routes should be generated. However, a drawback of this approach is that the number of generated feasible routes grows exponentially with the number of flights, which results in a significant increase in the model complexity. In this paper, in contrast to the set-partitioning approach that needs a sophisticated solution approach, we propose a formulation that uses a polynomial number of variables and constraints, which can easily handle real and large-scale problems. Secondly, it is also observed that most of the OAMRP models considered some maintenance constraints while neglecting the rest. To our best knowledge, the models by (Barnhart et al., 1998) and (Haouari et al., 2012) are the only models that considered three maintenance constraints (one maintenance visit every four days, maximum flying hours and maximum number of take-offs). However, these studies have not considered the workforce capacity constraints, except the model by (Haouari et al., 2012) who considered these constraints, however the authors relaxed them in their computational experiments. Imagine, if, for instance, the model neglects the workforce capacity and schedules four aircraft for maintenance in a station with insufficient workforce capacity. It is highly probable that the waiting time of some of the aircraft will be prolonged in receiving maintenance. This waiting time can be avoided if more hands and/or resources

are deployed to handle the excess traffic, resulting in additional cost being incurred for not considering the workforce capacity. On the other hand, it is observed that most of the OAMRP models do not pay attention to the working hours of the maintenance stations (e.g. opening and closing time). This results in aircraft arriving at times outside the working times of the maintenance stations, leading to a long waiting time for the aircraft till the maintenance station is operational. This situation causes cancellation of subsequent scheduled flights and additional cost will be incurred to recover these flights. Therefore, the viability of the workforce capacity and the working hours of the maintenance stations necessitate their addition to the OAMRP. Since our polynomial formulation is scalable compared to the set-partitioning formulations, all the operational maintenance requirements besides the workforce capacity and the working hours of the maintenance stations can be considered in one model. These features strengthen the applicability of the proposed model in actual practice. In addition to these contributions, we propose an efficient solution method, which generates high quality solutions for large-scale test instances in a short computational time, allowing the model to handle real situations in the airline industry.

3. The model

Given a schedule of flight legs, our proposed OAMRP aims to generate maintenance feasible routes to be flown by each aircraft. The objective function of the proposed model is to maximize the total potential profit, which is the difference between the through value and the penalty cost. Through value is the revenue that comes from the through connects which attract the passengers. On the other hand, the penalty cost is the extra cost to the airlines in neglecting the maintenance workforce capacity.

3.1 Modified connection network

The connection network is one of the commonly used networks for AMRP (Gopalan & Talluri, 1998; Haouari et al., 2012). The nodes of the network represent the flight legs, whereas, the arcs represent the possible connections between those flight legs. As mentioned earlier, the proposed OAMRP model considers all the maintenance constraints and determines when and where each aircraft should undergo maintenance operations. To do so, initially, the specific number of maintenance visits for each aircraft is determined based on the overall number of flying hours in the schedule, the available number of aircraft, and the maximum flying time for each aircraft. Then, we assign that specific number of maintenance visits to each aircraft in the fleet. Before each maintenance operation, the operational constraints should be considered and monitored. In order to keep covering the flight legs and assigning maintenance operations simultaneously, we need to add other arc and node types. Therefore, the connection network is slightly modified to include two types of nodes and three types of arcs, as shown in Figure 1. Starting with the node set, it includes the maintenance station node set (MT) besides the flight leg node set (NF). On the other hand, the arc set is modified to include another two arc sets, which are the maintenance arc set ($MAINT$) and the auxiliary arc set (AUX), besides the ordinary arc set (ORD). In the original connection network and our modified network, the ordinary arc $ord(i,j) \in ORD$ is used to connect flight legs i and j, connect flight legs and source node at the beginning of the route construction, and connect flight legs and sink node at the end of the route construction. On the other hand, in our modified network, the maintenance arc $maint(i,m) \in MAINT$ is incorporated in the network in order to connect flight leg i and maintenance station m, whereas the auxiliary arc $aux(m,j) \in AUX$ is added to connect maintenance station m and flight leg j. The auxiliary arc allows going back to use the ordinary arcs after finishing the maintenance operations.

One of the obvious question that might be asked is, “what is the interpretation of the source and sink nodes, and how does the connection network capture the origin airports and the destination airports”. Actually, the source and sink nodes are artificial or dummy nodes, and they are incorporated in the network not to represent any real aspect, such as the origin or the destination airports, but to help the mathematical model in initiating and completing the aircraft route construction (Başdere & Bilge, 2014). At the beginning, we formulate a constraint by using the source node to ensure route initiation for the aircraft, then, the aircraft move throughout the network nodes. Finally, to ensure route completion for the aircraft, we formulate

another constraint with the help of the sink node. To clarify how the connection network captures the origin airports and the destination airports, we need to mention here that there are two representations for the flight leg in the connection network (Liang & Chaovalitwongse, 2009). The first representation of the flight leg, as shown in the left-hand side of Figure 2, consists of three items; the origin airport node, the destination airport node, and the flight arc that connects the origin and destination nodes. Recently, to simplify this representation, the previous three items were replaced by a single node, which represents the flight segment or leg, including its origin airport, destination airport, arrival time, departure time, and flight duration (Başdere & Bilge, 2014). This simple representation, as shown in the right-hand side of Figure 2, motivates us to consider it in our connection network.

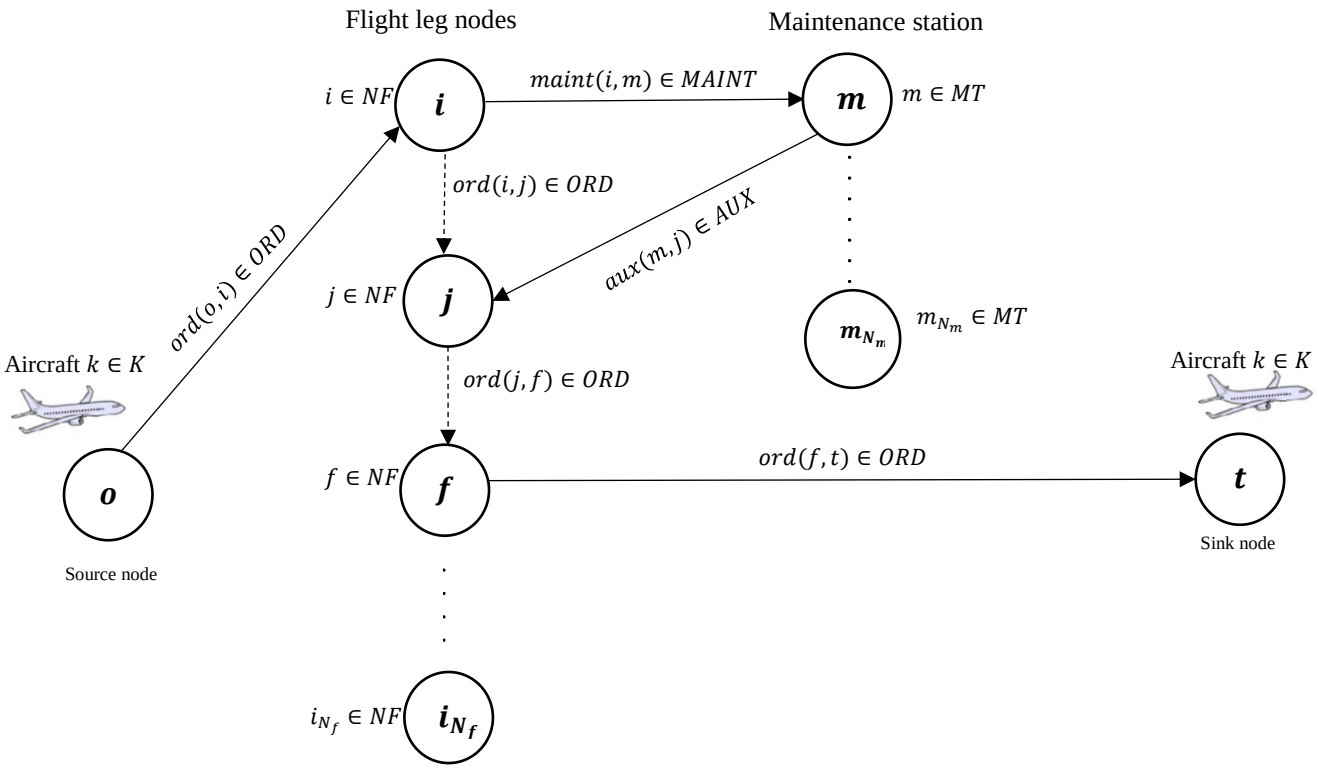

Figure 1: Modified connection network representation.

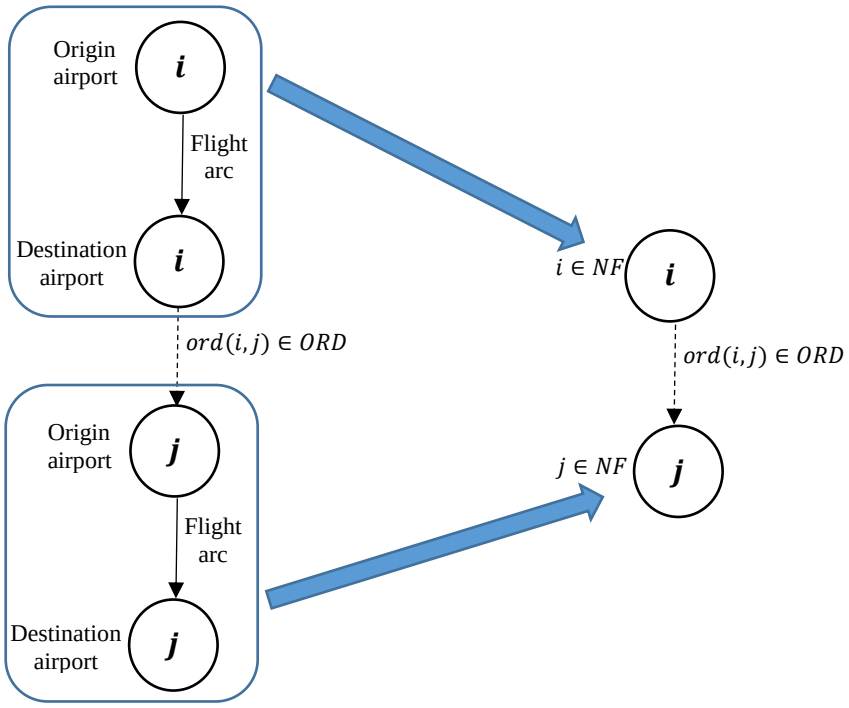

Figure 2: The two representations of the flight leg

3.2 The mathematical model formulation

In this section, a multi-commodity network flow based MILP formulation for our OAMRP is presented. In the network, each aircraft represents a separate commodity circulated under three main decision variables. These decision variables are x_{ijkv}, y_{imkv}, z_{mjkv}, which represent ordinary arcs, maintenance arcs, and auxiliary arcs, respectively. The decision variable $RTAM_{kv}$ is cast to determine when the auxiliary arcs can be used after finishing the maintenance operation. Each maintenance station has its own workforce capacity, and when the number of assigned aircraft to the maintenance station exceeds its workforce capacity, the airline has to pay a penalty cost to let the maintenance station call in extra hands so that the excess number of aircraft can be served. To handle this situation, the decision variable $ENOA_m$ is incorporated in the model to help in calculating the number of the aircraft that exceeds the maintenance workforce capacity, so that the penalty cost can be easily determined.

Before presenting the proposed model, we specify its scope as follows:

- The planning horizon of the proposed model is 4-day. This horizon is selected due the following reasons. Firstly, airlines tend to repeat flight schedules every 4 days, in order to ease the requirement of satisfying one maintenance visit every 4 days for the aircraft, as mandated by FAA. Secondly, the 4-day horizon is commonly used in the literature (Feo & Bard, 1989; Talluri, 1998). Thirdly, the 4-day horizon permits different flight schedules each day, whereas the daily horizon assumes repeating the flight schedule each day. Practically, airlines permit variations on the flight schedule each day to cope with the demand fluctuation of different flight legs (Eltoukhy et al., 2017b). Therefore, the 4-day horizon is more practical than the daily horizon in handling different flight schedules each day, so that our proposed model can handle real flight schedules. It should be noted that the 4-day horizon can be considered as a cyclic schedule in one case, if the same flights planned in the 4 days, with different flights from one day to another, are repeated every four days. Lastly, extending the planning horizon beyond 4-day, such as 5 or 6 days, or even weekly horizons, produces routes that are susceptible to disruptions. Therefore, based on the previous observations, we stick with the 4-day horizon to be consistent with the existing research and practice.
- The model only considers existing maintenance stations and there is no recommendation for constructing new stations.
- The maintenance stations are located in the hub airports.
- The number of workforce teams in each maintenance station is deterministic.
- All the maintenance operations discussed in this paper are Type A maintenance checks.

To formalize the representation of the proposed OAMRP, we first define the notations that are frequently used throughout this paper, before giving the detailed formulation. First, we start by listing the sets and the indices associated with each set as follows.

$i, j \in NF$:	Set of flight legs.
$k \in K$:	Set of aircraft.
$m \in MT$:	Set of maintenance stations.
$a \in A$:	Set of airports.
$v \in \{1,2,...,V\}$:	Average number of maintenance operations that should be performed on each aircraft during the planning horizon.
$\{o,t\}$:	Dummy source and sink nodes of the network.

Next, the parameters are defined as follows.

DT_i :	Departure time of flight leg i.
AT_i :	Arrival time of flight leg i.

TRT: Turn-around time, which is consumed for getting passengers off, unloading the luggage, changing the gate, boarding, loading the luggage, and fueling the aircraft.

O_{ia}: Origin binary indicator of flight leg i such that $O_{ia}=1$ if the origin of flight leg i and the airport a are the same, and 0 otherwise.

D_{ia}: Destination binary indicator of flight leg i such that $D_{ia}=1$ if the destination of flight leg i and the airport a are the same, and 0 otherwise.

FT_i: Flight duration of flight leg i. It should be noted that FT_i is different from $AT_i - DT_i$ due to the time difference between different countries as the departure and arrival times are given based on the local time at the location the event is taking place.

b_{ij}: Through value of the connection between flight legs i and j.

$T_{\max}$: Maximum flying time between two successive maintenance operations.

$C_{\max}$: Maximum number of take-offs between two successive maintenance operations.

MP_m: Number of workforce teams that is available in maintenance station m. Note that each team can serve only one aircraft. Therefore, MP_m can also reflect the maximum number of aircraft that can visit station m

OT_m: Opening time for maintenance station m.

ET_m: Closing time for maintenance station m.

Mb_{ma}: Maintenance binary indicator of maintenance station m such that $Mb_{ma}=1$ if the maintenance station m is located at airport a, and 0 otherwise.

MAT: Duration of a Type A maintenance check. Usually, this ranges from 7 to 9 hours as it involves visual inspection of major systems such as landing gear, engines and control surfaces (Haouari et al., 2012; Başdere & Bilge, 2014). Based on this observation, we assume that $MAT=8$ hours. It should be pointed out that this assumption is consistent with the data collected from EgyptAir.

KT: Total number of aircraft used to cover the flight legs.

V: The average number of maintenance operations that should be performed on each aircraft, which is calculated by using the following rule; $V=\sum_{i\in NF} FT_i / (T_{\max} KT)$

M: a considerably large number.

PC_m: Penalty cost paid by the airline for each aircraft assigned to maintenance station m, such that the aircraft assignment exceeds the maintenance workforce capacity of the maintenance station.

The decision variables are:

$x_{ijkv} \in \{0,1\}$: =1 if flight legs i and j are covered by aircraft k before performing maintenance operation number v and 0 otherwise.

$y_{imkv} \in \{0,1\}$: =1 if aircraft k covers flight legs i then perform maintenance operation number v at maintenance station m and 0 otherwise.

$z_{mjkv} \in \{0,1\}$: =1 if aircraft k covers flight legs j after performing maintenance operation number v at maintenance station m and 0 otherwise.

$RTAM_{kv} > 0$: Time an aircraft k completes maintenance operation number v and is able to resume covering the flight legs. This decision variable is not required to be an integer as the time might be fractional.

$ENOA_m \geq 0$: Number of aircraft that exceeds the workforce capacity of maintenance station m.

Based on the above notations, the mathematical model of OAMRP can be written as follows:

Model 1

$$Max \sum_{k\in K}\sum_{i\in NF}\sum_{j\in NF}\sum_{v\in V} b_{ij}x_{ijkv} - \sum_{m\in MT} ENOA_m PC_m \tag{1}$$

s.t.

$$\sum_{k\in K}\left(\sum_{j\in NF\bigcup\{t\}}\sum_{v\in V} x_{ijkv} + \sum_{m\in MT}\sum_{v\in V} y_{imkv}\right) = 1 \qquad \forall i \in NF \tag{2}$$

$$\sum_{j\in NF} x_{ojkv} + \sum_{m\in MT} y_{omkv} = 1 \qquad \forall k \in K, \forall v \in V \tag{3}$$

$$\sum_{i\in NF} x_{itkv} + \sum_{m\in MT} z_{mtkv} = 1 \qquad \forall k \in K, \forall v \in V \tag{4}$$

$$\sum_{j\in NF\bigcup\{o\}} x_{jikv} + \sum_{m\in MT} z_{mikv} = \sum_{j\in NF\bigcup\{t\}} x_{ijkv} + \sum_{m\in MT} y_{imkv} \qquad \forall i \in NF, \forall k \in K, \forall v \in V \tag{5}$$

$$\sum_{j\in NF}\sum_{v\in V} y_{jmkv} = \sum_{j\in NF\bigcup\{t\}}\sum_{v\in V} z_{mjkv} \qquad \forall m \in MT, \forall k \in K \tag{6}$$

$$AT_i + TRT - DT_j \leq M(1 - x_{ijkv}) \qquad \forall i \in NF, \forall j \in NF, \forall k \in K, \forall v \in V \tag{7}$$

$$\sum_{k\in K} x_{ijkv} \leq \sum_{a\in A} D_{ia}O_{ja} \qquad \forall i \in NF, \forall j \in NF, \forall v \in V \tag{8}$$

$$AT_i + MAT - ET_m \leq M(1 - y_{imkv}) \qquad \forall i \in NF, \forall m \in MT, \forall k \in K, \forall v \in V \tag{9}$$

$$OT_m - AT_i \leq M(1 - y_{imkv}) \qquad \forall i \in NF\bigcup\{o\}, \forall m \in MT, \forall k \in K, \forall v \in V \tag{10}$$

$$\sum_{k\in K} y_{imkv} \leq \sum_{a\in A} D_{ia}Mb_{ma} \qquad \forall i \in NF, \forall m \in MT, \forall v \in V \tag{11}$$

$$\sum_{k\in K} z_{mjkv} \leq \sum_{a\in A} Mb_{ma}O_{ja} \qquad \forall m \in MT, \forall j \in NF, \forall v \in V \tag{12}$$

$$RTAM_{kv} \geq \sum_{i\in NF\bigcup\{o\}}\sum_{m\in MT}\left(AT_i + MAT\right)y_{imkv} \qquad \forall k \in K, \forall v \in V \tag{13}$$

$$RTAM_{kv} - DT_j \leq M(1 - z_{mjkv}) \qquad \forall m \in MT, \forall j \in NF, \forall k \in K, \forall v \in V \tag{14}$$

$$\sum_{i \in NF \cup \{o\}} \sum_{j \in NF} x_{ijkv} \le C_{\max} \qquad \forall k \in K, \forall v \in V \tag{15}$$

$$\sum_{i \in NF \cup \{o\}} \sum_{j \in NF} FT_j x_{ijkv} \le T_{\max} \qquad \forall k \in K, \forall v = 1 \tag{16}$$

$$\sum_{i \in NF} \sum_{j \in NF} FT_j x_{ijkv} + \sum_{m \in MT} \sum_{j \in NF} FT_j z_{mjkv} \le T_{\max} \qquad \forall k \in K, \forall v \in V / \{1\} \tag{17}$$

$$\sum_{i \in NF} \sum_{m \in MT} \sum_{v \in V} y_{imkv} \ge 1 \qquad \forall k \in K \tag{18}$$

$$\sum_{i \in NF \cup \{o\}} \sum_{k \in K} \sum_{v \in V} y_{imkv} \le MP_m \qquad \forall m \in MT \tag{19}$$

$$ENOA_m \ge \sum_{i \in NF \cup \{o\}} \sum_{k \in K} \sum_{v \in V} y_{imkv} - MP_m \qquad \forall m \in MT \tag{20}$$

$$x_{ijkv} \in \{0,1\} \qquad \forall i \in NF, \forall j \in NF, \forall k \in K, \forall v \in V \tag{21}$$

$$y_{imkv} \in \{0,1\} \qquad \forall i \in NF, \forall m \in MT, \forall k \in K, \forall v \in V \tag{22}$$

$$z_{mjkv} \in \{0,1\} \qquad \forall m \in MT, \forall j \in NF, \forall k \in K, \forall v \in V \tag{23}$$

$$RTAM_{kv} > 0 \qquad \forall k \in K, \forall v \in V \tag{24}$$

$$ENOA_m \ge 0 \qquad \forall m \in MT \tag{25}$$

The objective function (1) is the maximization of the total profit, which is the through value (revenue) minus the total penalty cost. Note that the total penalty cost depends on two terms. Firstly, the number of aircraft that exceeds the maintenance workforce capacity of station m, known as $ENOA_m$. Secondly, the cost paid for each aircraft that exceeds the maintenance workforce capacity of station m (known as PC_m). For $ENOA_m$, it is determined according to constraints (20). For PC_m, it is assessed practically as follows. When the maintenance station experiences excess traffic, it tries to avoid significant delay for that traffic by assigning overtime for the teams serving in different shifts. In other words, when the excess traffic occurs during the daytime shift, the maintenance station might call in the teams that will serve during the overnight shift, and vice versa. It should be noted that each aircraft can be served by only one called team. For example, if the maintenance station has two aircraft as excess traffic, it needs to call in two teams. Based on this observation, the number of called teams reflects the number of aircraft that exceeds the maintenance workforce capacity, so the number of called teams can be expressed as $ENOA_m$. These called teams should be compensated for their overtime load. For this purpose, the maintenance station imposes an extra cost on the airline to cover the cost of the overtime load. This extra cost is called the penalty cost. To assess the value of penalty cost PC_m, we need to check whether the team, which serves the aircraft, is called from the daytime shift to the overnight shift or vice versa. Discussion with experts at EgyptAir revealed that the variation of PC_m is usually not significant. Based on this

observation, for simplicity, we assume that PC_m is fixed. To summarize, this penalty cost reflects the cost paid by the airline to the maintenance station as compensation for the called workforce teams so that the excess aircraft traffic can receive maintenance without significant delay. Constraints (2), (3), and (4) describe the coverage constraints. Constraints (2) indicate that each flight leg must be covered exactly by one aircraft. The constraints in (3) ensure that each aircraft starts its route, whereas constraints (4) guarantee the route completion.

In order to keep the circulation of the aircraft throughout the network, the balance constraints (5) and (6) are formulated. Constraints (5) keep the balance when aircraft covers the flight leg nodes. These constraints indicate that if the aircraft covers the flight leg either by using the ordinary arc or the auxiliary arc, then the next flight must be covered either by using the ordinary arc or the maintenance arc. Same as constraints (5), constraints (6) keep the balance when the aircraft visits the maintenance station. Constraints (6) ensure that if the aircraft covers the flight leg and visits the maintenance station by usage of the maintenance arc, then the aircraft must leave the maintenance station and cover the next flight by using the auxiliary arc.

In order to connect two flight legs by using same aircraft via the ordinary arc, that connection should be feasible in terms of time and place considerations, as described by constraints (7) and (8). Constraints (7) indicate the time constraints such that the aircraft can cover two successive flight legs, if the arrival time of the first one plus the turn-around time is less than or equal to the departure time of the second one. To clarify constraints (7), we give a simple example. Suppose that we have three flight legs; the first flight arrives an airport at $AT_1 = 4$ pm, and the others depart the same airport at $DT_2 = 2$ pm, and $DT_3 = 7$ pm. We have one aircraft that should connect these flight legs, given that the $TRT = 1$ hour. In this case, we have two scenarios, connecting the first and the second flights, or connecting the first and the third flights. Regarding the first scenario, if we apply constraints (7), we will find that left-hand side of the inequality is positive, therefore, the right-hand side should be larger. This can be achieved only when $x_{ijkv} = 0$, which makes the right-hand side equals the large number *M*. So, for this scenario, the connection is not valid as the arrival time of the first flight plus the turn-around time is larger than the departure time of the second flight. Regarding the second scenario, it is the opposite of the first scenario, in which the connection is valid. Regarding the place constraints in (8), they ensure that the aircraft can cover two consecutive flight legs, if the destination of the first one and the origin of the second one are the same.

To prepare a maintenance visit for the aircraft after covering a flight leg using the maintenance arc, we should consider the place and time issues for the last covered flight leg and the potentially visited maintenance station. These considerations are summarized by constraints (9), (10), and (11). Constraints (9) and (10) describe the time issue by considering the working hours of the maintenance stations. Constraints (9) guarantee that the aircraft can visit the maintenance station, if the arrival time of the last covered flight leg plus the duration of Type A maintenance check, is less than or equal to the closing time of the maintenance station. Similarly, Constraints (10) ensure that the aircraft can visit the maintenance station, if the arrival time of the last covered flight leg, is larger than or equal to the opening time of the maintenance station. Note that the arrival time for the source node is zero. The place constraints in (11) ensure that the aircraft can visit the maintenance station, if the destination airport of the last covered flight leg and the location of maintenance station are the same.

After finishing the maintenance operation, the aircraft should move from the maintenance station and cover the next flight leg by using the auxiliary arc. For this purpose, constraints (12), (13), and (14) are cast, which represent the time and place considerations for the maintenance station and the next flight to be covered. Constraints (12) constitute the place constraints such that the aircraft can cover the next flight leg after performing the maintenance, if the origin airport of that flight leg and the location of the maintenance station are the same. The time constraints in (14) guarantee that the aircraft can cover the next flight leg after the maintenance operation, if the departure time of the next flight leg is larger than or equal to the ready time of aircraft $RTAM_{kv}$, which is determined according to constraints (13).

It should be noted that the coverage and balance constraints do not force an aircraft that needs maintenance to undergo maintenance operations. Therefore, the operational restrictive constraints (15), (16), (17), and (18) are cast. Constraints (15) guarantee that the number of take-offs between maintenance operations does not exceed the maximum number of take-offs. Similarly, constraints (16) and (17) are the restrictive constraints regarding the accumulated flying times between maintenance operations. Constraints (18) are formulated to ensure that the number of maintenance visits by each aircraft is larger than or equal to one. Since the planning horizon in our study is 4-day, and constraints (18) ensure that the number of maintenance visits for each aircraft is larger than one, so the first operational maintenance constraint (one visit every four days) is satisfied.
To prepare appropriate maintenance visits for the aircraft, it is very important to check whether the maintenance station has sufficient workforce capacity or not. Therefore, the workforce capacity constraints are cast in constraints (19), to avoid the overcapacity problem, ensuring that the number of maintained aircraft does not exceed the maintenance workforce capacity. To calculate the number of aircraft that exceeds the maintenance workforce capacity, the constraints (20) are cast. Finally, the constraints (21) - (25) define the domain of the decision variables.

The main differences between model 1 and existing models in the literature are as follows:

- Consideration of the workforce capacity of the maintenance stations, as shown by Eq. (19), and inclusion of the penalty cost for assigning aircraft to a maintenance station without sufficient workforce capacity. This workforce capacity consideration was neglected in the model of (Barnhart et al., 1998), and was relaxed in the computational experiments by (Haouari et al., 2012). This would cause assigning aircraft to maintenance stations with insufficient workforce capacity, resulting in prolonging the waiting time for the aircraft. This can be avoided if more hands and/or resources are deployed to handle the excess traffic, resulting in an additional penalty cost. So, considering workforce capacity helps airlines to avoid that situation, which results in a reduction in the penalty cost.
- Consideration of the working hours of the maintenance stations, as shown by Eqs. (9) and (10). This point has not received attention in the previous models (Sriram & Haghani, 2003; Sarac et al., 2006; Haouari et al., 2012; Başdere & Bilge, 2014). Ignoring this consideration results in a long waiting time for the aircraft in receiving maintenance service, as the aircraft might arrive at a time outside the working hours of the maintenance station. This situation can cause cancellation of subsequent scheduled flights and additional cost will be incurred to recover these flights. So, considering this point helps airlines to avoid flight cancelations, leading to a reduction in the operational costs.
- Most of the models assumed, for simplicity, that the maintenance operations are only carried out overnight (Liang et al., 2011; Sriram & Haghani, 2003). In contrast to this assumption, in reality, the aircraft can receive maintenance during the working hours of the maintenance stations, which covers 24 hours of the day, including the daytime shift and the overnight shift. If we follow the assumption that maintenance operations are only carried out overnight, while neglecting the daytime shift, it means that the aircraft that need maintenance, when they arrive at night, they can receive maintenance during the overnight shift without any delay. Meanwhile, when the aircraft arrive at morning, they will be stuck at the airport till night before receiving maintenance operations. This results in a long waiting time for these aircraft, so that subsequent flights to be covered by the aircraft will be cancelled. To avoid this situation, we consider the working hours of the maintenance stations, which helps us to add the daytime shift besides the overnight shift. So, the aircraft that arrive in the morning can receive maintenance during the daytime shift. This results in avoiding a long waiting time till the overnight shift, leading to a reduction in the number of cancelled flights.

3.3 Complexity analysis

In the literature, the complexity of the MIP models can be expressed based on the number of decisions variables and constraints (Liang et al., 2011; Dong et al., 2016). Using such an approach reveals that, in our model, the number of decision variables is $|K|\times|V|\times(|NF|^2+2|NF|\times|MT|+1)+|MT|$, whereas the generated constraints are at most $O(|NF|^2\times|K|\times|MT|\times|V|)$. For more clarification about our model scale, we describe an example using the smallest test instance in this paper. This test instance includes 40 flight legs, 8 aircraft, 4 maintenance stations, and each aircraft is maintained twice. To handle this test instance using our model, there are $8*2*(40^2+2*40*4+1)+4=30740$ decision variables, and the number of constraints is at most $O(40^2*8*4*2)$.

Based on the previous description, the model space complexity (number of decision variables) can be expressed as $O(|K|\times|V|\times|NF|^2)$. These decision variables help $|NF|$ flight legs to be covered by $|K|$ aircraft, exactly once, and each aircraft visits maintenance station $|V|$ times. This is the typical description of the partition problem (Sarac et al., 2006; Başdere & Bilge, 2014). Since our model includes a partition problem that is known as NP-complete, our model is NP-hard.

To demonstrate the scale advantage of our model formulation, we compare the space complexity of our model $O(|K|\times|V|\times|NF|^2)$ with that of set-partitioning partitioning formulation $O(2^{|NF|})$. From this comparison, we can see that our polynomial model formulation is more scalable than the set-partitioning formulation. This is because the number of decision variables in our model is much less than the possible number of feasible routes generated by set-partitioning formulation, which can reach up to a few million for the smallest test instance used in this paper.

Through comparison, our polynomial model formulation has a significant advantage in terms of model scalability, which is very important in practical implementation. However, it does not mean that our model can be directly handled by using commercial optimization software like CPLEX in a reasonable computational time. Our preliminary results reveal that the small size test instances can be efficiently solved by CPLEX. However, CPLEX fails to even find a feasible solution for medium and large-scale test instances. Therefore, we develop an efficient algorithm to solve real and large-scale problems in a reasonable computational time, as described in the following section.

4. Solution approach

Before describing our solution method, we present the solution methods used to solve the existing OAMRP. In the literature, there are two main solution methodologies. Firstly, formulating the OAMRP model as a set-partitioning problem and applying column generation to solve the model, as shown by (Sarac et al., 2006). Solving our model using column generation requires formulating the model as a set-partitioning model, in which all the possible feasible routes should be generated. Since the number of generated feasible routes of the set-portioning formulation grows exponentially with the number of flight legs, it is challenging to solve medium and large-size test instances in a reasonable computational time. As we mentioned earlier, our target is to solve real and large-size test cases, therefore, using column generation is not appropriate for adoption in our study. This is confirmed through the work by (Sarac et al., 2006), where the authors used column generation as a solution method, which could not solve large test cases and only handled a small test case with 175 flight legs. The second solution methodology is in formulating the OAMRP model as a multi-commodity network flow model and applying different tools to solve the model, as shown by (Sriram & Haghani, 2003), (Haouari et al., 2012) and (Başdere & Bilge, 2014). Starting with the study by (Sriram & Haghani, 2003), the authors proposed two models. The first model was solved by designing an effective algorithm which was a combination of depth search and random search, but it could not handle large-scale test instances and it handled only small test cases with 58 flight legs. Applying that algorithm to solve our model is not useful since our target is to solve real and

large-scale test problems that contain up to 400 flight legs. It should be noted that EgyptAir, that provided the data for our study, usually offers around 120 different flight legs. This number enables EgyptAir to operate around 400 flight legs every four days. The second model by (Sriram & Haghani, 2003) was only proposed but not yet solved. It is noteworthy that a second model that was not yet solved is even more simple compared to that one proposed in our study. It is simple because it neglects the maximum number of take-offs between two successive maintenance operations and the minimum number of visits within four days, resulting in reduction in the number of constraints and decision variables, whereas our model considers all the operational constraints. Moving to the work by (Haouari et al., 2012), their proposed model used CPLEX 12.1, which is also adopted to solve our model, but only for small size test instances. Lastly, (Başdere & Bilge, 2014) solved their model by using B&B for small-scale test instances, whereas the large-scale instances were handled by using CA. Since using B&B is time consuming and sometimes fails to provide feasible solutions for medium and large-scale problems, selecting B&B is not a promising idea. On the other hand, using CA is a good idea for solving simple models, but when the model's variables and constraints increase, as in our model, it is difficult to use CA as a solution method.

From the above list, we can see that most of the OAMRP models in the literature were relatively easier to be solved because these models were formulated as multi-commodity network flow models and the number of decision variables and constraints were relatively low, which made these models easy to solve. However, it is much trickier to solve our proposed model due to two points: (1) considering all the operational considerations add more decision variables and constraints to the model. The model is a multi-commodity network flow model with side constraints, which is NP-hard, so the computational time is expected to be long as the number of decision variables and constraints increase. (2) the original structure of the multi-commodity network flow model can be reflected by the decision variable x_{ijkv} and its corresponding constraints, as described by Eqs. (2)- (5), (7) and (8). This original structure of multi-commodity network flow model can be easily handled. Since our model considers all the operational requirements, new decision variables ($y_{imkv}, z_{mjkv}, RTAM_{kv},$ and $ENOA_m$) are created in the model, resulting in new terms being added to Eqs. (2)- (5). Meanwhile, new constraints are cast to indicate all operational considerations, and to force the aircraft to undertake the maintenance operation, as described by Eqs. (6) and (9)- (20). Actually, these decision variables and maintenance constraints seriously destroy the original structure of the multi-commodity network flow model and make it difficult to be solved. These two points are confirmed through the research work by (Sriram & Haghani, 2003). The authors tried to use the most popular random search method, namely, genetic algorithms as a solution method. It was discovered that genetic algorithms produced poor solution quality in a long computational time, especially for medium and large-scale test instances.

Based on the above observations, we propose an efficient solution algorithm for solving model 1. We noticed from the model structure that it is difficult to build routes that maximize the profit and satisfy all the maintenance requirements simultaneously. Therefore, our algorithm, first, prepares sub-routes that maximize the profit, while considering the coverage, balance, time and place constraints, as shown by Eqs. (2) -(8). Then, second, the algorithm keeps trying to construct complete routes using the pre-determined sub-routes, while considering all the maintenance constraints described by Eqs. (9) -(20). The steps of the algorithm are explained as follows:

Step 0: Prepare a list that represents the aircraft (K) and make another list to represent the flight leg nodes (NF).

Step 1: Determine the average number of maintenance operations (V) that should be performed on each aircraft in the fleet by using the following rule:

$$V = \sum_{i \in NF} FT_i / (T_{\max} KT) \qquad (26)$$

Step 2: Split the list of NF into two lists. The first list is called the star list (SL), and contains the through connects, thus this list is given high priority during the route construction. The second list is called the normal list (NL). Since NL contains the remaining flight legs, it is given low

priority during route construction. To do the split, for each pair of flight legs in the NF list, the connecting time between each pair is calculated. If the connecting time of the pair has a through value, then this pair is a through connect. Therefore, this pair should be stored in SL and should be removed from NF. Otherwise, store the rest of the flight legs in NL.

Step 3: Use SL to construct another list called the sub-routes list (SRL). Each sub-route is constructed by connecting two pairs from SL. The two pairs can be connected, especially when the ending flight of the first pair and the starting flight of the second pair are the same. Each constructed sub-route is stored in SRL. Of course, not all the pairs can be connected, so the remaining non-connected pairs should be stored in SRL. So, by the end of this step, we have three lists, K, SRL, and NL.

Step 4: Initialize the number of iterations=1.

Step 5: Pick the kth aircraft from the K list. If the list is empty, then go to step 8, otherwise go to step 6.

Step 6: Start to construct the complete route for the kth aircraft by using backward and forward insertion approaches, while considering the maintenance constraints shown in Eqs. (9) -(20). In order to conduct the backward insertion approach, we follow the following sub-steps:

- Step a: Pick one element from SRL randomly, because it contains pairs with high priority. If SRL is empty, then low priority NL is used for picking that element. This picked element is considered the first part of the constructed route.
- Step b: For the picked element, identify the starting flight leg, which is either the first flight leg if the element is picked from SRL, or it is the element itself if it is selected from NL.
- Step c: Search for suitable elements to be inserted backwardly to (before) the picked element. These elements might be either sub-routes from SRL or flight legs from NL. Firstly, we scan through SRL due to its high priority. As we mentioned earlier, if SRL is empty, we use the second option by scanning through NL. The scan is conducted while considering constraints described by Eqs. (7) and (8). If both SRL and NL are empty, then go to step i, otherwise go to step d.
- Step d: Identify the list of potential elements. Then, calculate the connecting time and the corresponding through value for each potential element. In the case of no potential elements, then go to step h, otherwise go to step e.
- Step e: Select the element with the highest through value. Then, check whether the maintenance constraints stated in Eqs. (15) -(18) are violated or not. If these constraints are violated, then go to step f, otherwise go to step g.
- Step f: Prepare an appropriate maintenance visit by considering the working hours and location constraints of the maintenance stations stated in Eqs. (9) -(11), and the workforce capacity constraints described by Eq. (19). After finishing the maintenance operation, pick suitable element from SRL or NL, while considering the constraints described by Eqs. (4) -(6) and Eqs. (12) -(14). Then, go to step b.
- Step g: Add the selected element to the route and remove that element from SRL or NL.
- Step h: Update the starting leg, then go to step c.
- Step i: Terminate the backward insertion approach.

After conducting the backward insertion, we start conducting the forward insertion approach. First, we identify the ending flight leg for the element picked in step b. The ending flight leg is either the last flight leg if the element is picked from SRL, or it is the element itself if it is selected from NL. Second, we follow the same steps (c)-(i) but insert the suitable elements forwardly to the element that is picked in step a.

Step 7: Set the end to the route constructed for the kth aircraft, remove the aircraft k from the K list, and go to step 5.

Step 8: Calculate the solution of the current iteration, compare with the solution of the previous iteration, and save the best solution.

Step 9: Check whether the stopping criteria is satisfied or not. If it is not satisfied, then increase the iteration number, update the empty lists of K, SRL, and NL by using the same lists produced by step 3, and go to step 5. If the stopping criteria is satisfied, then terminate the algorithm.

Figure 3 presents flow chart of this solution algorithm procedure. This procedure is iterated until satisfying the stopping criteria, which are as follows: (1) the current solution reaches the exact solution, while handling small-scale test instances. (2) the current solution reaches the best upper bound, while handling medium and large-scale test instances. (3) the number of iterations exceeds the maximum number of iterations. In all test instances, the maximum number of iterations is set at 1000.

One of the obvious questions after using the solution algorithm is how to evaluate the performance of the proposed solution algorithm. To make this evaluation, we propose comparing the solution obtained from our efficient algorithm with the optimal solution generated by CPLEX, especially for small size test instances. In the case of medium and large size test instances, we propose using the best upper bound (UB) obtained from CPLEX as a criterion to assess the performance of the proposed solution algorithm, since CPLEX fails to even produce a feasible solution within reasonable computational time. To obtain the UB, we set the maximum CPU time for CPLEX to be 6 hours, since longer time does not provide a better bound.

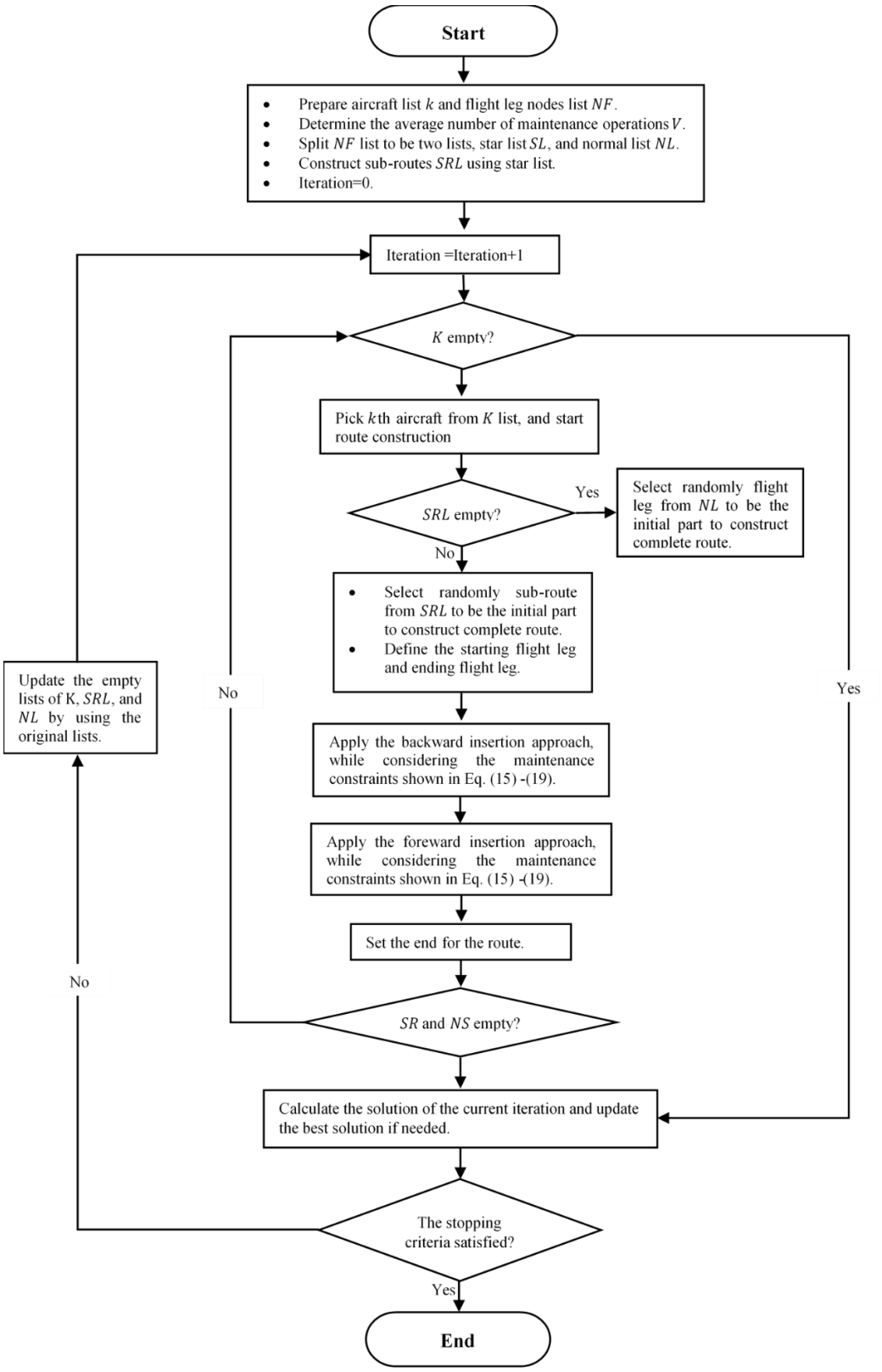

Figure 3: Flowchart of the solution algorithm

5. Comparison between model 1 and models in the literature

To test the implications on profitability after considering the constraints of the workforce capacity and the working hours of maintenance stations, a comparison between our model (model 1) and existing models was conducted. For this purpose, we modified model 1 to be similar to the models in the literature, such as the work by (Haouari et al., 2012) and (Başdere & Bilge, 2014). We call the modified model, model 2, which considers the same objective function and constraints as model 1, except relaxing the constraints of the working hours and the workforce capacity of the maintenance stations, as in Eqs. (9), (10), (13), and (19).

To solve model 2, we applied the same algorithm presented in section 4 with a small modification, since it solves model 1 efficiently. This modification includes relaxing the constraints of the workforce capacity and the working hours of the maintenance stations, throughout the whole algorithm.

6. Computational results

In this section, we present the computational results obtained from the proposed algorithm and those obtained directly from solving the MILP formulation. Note that the MILP formulation is solved using CPLEX 12.1, which is a common commercial optimization software. Since the nature of the OAMRP is combinatorial and the computational time increases enormously even for medium size test instances, it is difficult to solve all the test problems to optimality using commercial software, such as CPLEX and LINDO. Therefore, we were not able to obtain exact solutions for all the test problems. This computational study aims to verify the effectiveness of the proposed solution algorithm while solving real and large OAMRP. The experiments of this study were carried out using real flight schedule data sets from EgyptAir.

6.1 Test instances and experimental setup

The test instances used in our experiments contain ten real cases acquired from EgyptAir. In particular, our ten cases are constructed by extracting ten flight schedules in which each schedule is covered by a different fleet. Detailed information about the test instances are presented in Table 2.

For all test instances, EgyptAir stated that the turn-around time TRT is 45 minutes, the maximum flying time T_{max} is 40 hours, and the time required for the maintenance operation is 8 hours. Also, it is assumed by EgyptAir that the through value occurs if the connecting time between two consecutive flight legs, covered by the same aircraft, is between 45 minutes and 1.5 hour. In this study, all the through values are provided by EgyptAir. Also, the penalty cost is assumed to be 500 for each aircraft.

Before conducting our experiments, it should be noted that, to assess the average performance of the proposed algorithm, the runs of the proposed algorithm should be replicated several times. For this purpose, for all test cases, our algorithm runs are replicated thirty times, as any additional replications do not bring better results. All the test cases were carried out on an Intel i7 2.50 GHz laptop with 8 GB of RAM memory running on the Windows 10 operating system. All algorithms in this study were coded in MATLAB R2014a.

Table 2: Characteristics of all test cases.

Test cases	Number of flight legs	Fleet size	Maximum number of take-offs	Number of airports	Maintenance Stations
Case 1	40	8	10	4	4
Case 2	48	7	7	5	4
Case 3	64	8	7	7	4
Case 4	96	14	10	13	6
Case 5	120	13	10	8	6
Case 6	160	11	15	10	6
Case 7	200	15	15	8	9
Case 8	240	26	15	19	9
Case 9	296	30	15	26	9
Case 10	400	42	15	28	18

6.2 Performance characteristics while solving small size test instances

Table 3 shows the comparison of the results obtained from CPLEX and the proposed solution algorithm replications, while solving the first eight cases presented in Table 2. The first two columns of the Table represent the results of CPLEX, which are the exact or optimal solution (Z^*) and the computational time $CPU(s)$. The remaining columns of the Table summarize the results of the proposed algorithm. The Z_{best} column reports the best solution of the proposed algorithm replication, whereas the $\bar{Z}$ and σ_z columns represent the average and standard deviation summaries of the replication results. The $\overline{CPU(s)}$ column records the average computational time. It should be noted that $\overline{CPU(s)}$ is not the average time to find the best solution, as the best solution is reported after finishing the thirty replications. The computational time of the proposed algorithm is obtained from the MATLAB's internal calculation function. In order to assess the performance of the proposed algorithm, we use the optimality gap (%Difference) as a performance indicator. %Difference is computed by $(Z^* - \bar{Z})/ Z^*$.

Table 3: performance characteristics of CPLEX and proposed algorithm while solving small size cases

Test cases	**CPLEX**		**Proposed algorithm**				**%Difference**
	Z^*	$CPU(s)$	Z_{best}	$\bar{Z}$	σ_z	$\overline{CPU(s)}$	
Case 1	16,667	1.44	16,667	16,667	0	0.28	0
Case 2	2,333	3.08	2,333	2,333	0	0.30	0
Case 3	5,333	18.19	5,333	5,333	0	0.25	0
Case 4	10,000	53.06	10,000	10,000	0	0.26	0
Case 5	15,000	243.80	15,000	14,909	62.27	0.84	0.61
Case 6	22,000	372.05	22,000	21,852	95.92	1.56	0.67
Case 7	42,667	633.88	42,667	42,542	139.70	1.61	0.29
Case 8	34,083	9130.19	34,083	33,899	183.45	2.64	0.54

By looking at the results in Table 3, we can see that Z_{best} and $\bar{Z}$ of the proposed algorithm are equal to the $Z^*$for the first four cases. As the number of the flights and aircraft increase, Z_{best} still equals Z^*, but $\bar{Z}$ deviates from Z^* by at most 0.67%, as shown in the last four cases. By looking at the standard deviation, we note that there is no solution variability for the first four cases, but this variability slightly increases for the rest of cases. This confirms the stability and reliability of the proposed algorithm.
The computational time for both approaches in Table 3 reveals that the proposed algorithm is much faster than CPLEX since it produces the solution within, at most, 3 seconds while CPLEX needs up to 2.5 hours to solve the same problem, as shown in case 8. Figure 4 shows a comparison of the computational time for all eight cases while using the two approaches.

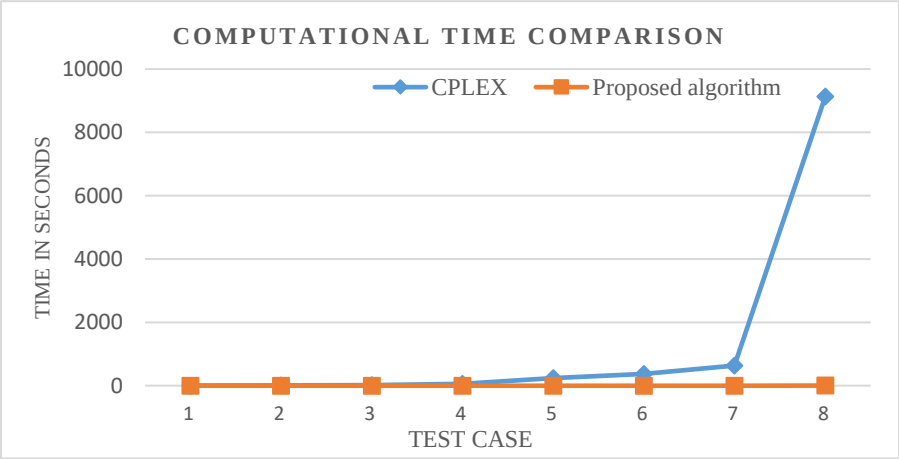

Figure 4: Comparison of the computational time while using CPLEX and proposed algorithm

So, it is clear from the results discussed in this section that, for small sized test instances, the proposed algorithm outperforms CPLEX in terms of computational time, showing significant time saving. Regarding the solution quality, the proposed algorithm produces high quality solutions since the average solution deviates from the optimal solution by at most 0.67% and the best solution always equals the optimal solution.
Solving test cases with sizes of up to 240 flights and 26 aircraft, is not large-scale enough to discuss the efficiency of the proposed algorithm; however, with the small size test instances, we are able to compare the performance with the exact methods as shown in this section. In the medium and large size test instances, CPLEX fails to even produce a feasible solution due to the size of problem, so computing the optimality gap becomes immeasurable. Therefore, comparing the proposed algorithm with CPLEX in large-scale instances is not meaningful as the comparison always favors the proposed algorithm. On the other hand, testing the proposed algorithm in large size test instances is necessary to show its applicability to handle real life problems. For this purpose, we perform computational experiments using medium and large size test instances in section 6.3. To evaluate the performance of the proposed algorithm while solving the medium and large cases, we propose using the gap (%GAP), which is the difference between the best upper bound (UB) obtained from CPLEX and the average solution obtained from the proposed algorithm.

6.3 Performance characteristics while solving medium and large size test instances

In this section, the proposed algorithm is tested on larger instances to assess its applicability and scalability to solve real life problems. The experiments in this section are carried out by using cases 9 and 10. It should be noted that CPLEX was run for 6 hours to obtain the UB. The summary of the proposed algorithm results can be seen in Table 4, including the same statistics as in Table 3.
We can see from Table 4 that the performance of the proposed algorithm still produces high quality solutions in a reasonable computational time. Starting with the solution quality, it is noted that Z_{best} reaches UB in all cases, whereas $\bar{Z}$ deviates from UB with a %GAP of around 0.66%. It is worthy of note that the %GAP produced in all cases is less than 0.7%. Regarding the computational time, clearly the proposed algorithm is very fast. For the largest test case, which is case 10, it is solved in few seconds. These experiments show that the proposed algorithm can be used efficiently to solve real life problems, as it handles large size test instances and provides profitable solutions in a very short computational time. By looking at the standard deviation, it indicates low solution variability, which confirms the stability and reliability of the proposed algorithm even when solving large size test instances.

Table 4: performance characteristics of the proposed algorithm while solving medium and large size cases

Test cases	UB	**Proposed algorithm**				**%GAP**
		Z_{best}	$\bar{Z}$	σ_z	$\overline{CPU(s)}$	
Case 9	60,333	60,333	59,997	325.77	2.57	0.55
Case 10	72,583	72,583	72,097	448.00	9.22	0.66

6.4 Performance analysis

In the previous sub-sections, we present the performance of the proposed algorithm while solving real life test instances. Presenting the performance of the proposed algorithm is insufficient to demonstrate its advantage over the existing solution methods in the literature. For this purpose, we conducted experiments to compare the performance of the proposed algorithm with another two existing solution methods, called CPLEX and compressed annealing (CA) that appeared in the work by (Haouari et al., 2012) and (Başdere & Bilge, 2014), respectively. CA is selected for this comparison because it showed a good performance in solving large-scale test instances, as reported by (Başdere & Bilge, 2014). So, it provides a good test for our proposed algorithm.
As described in section 3, our model 1 has more features than existing models in the literature. So, allowing model 1 to be solved by the above mentioned three methods will favour our proposed algorithm,

as our algorithm is tailored to solve model 1. Therefore, for fair comparison, we use model 2 to be solved by the three methods because model 2 captures the same features as the existing models in the literature. For our proposed algorithm, it was modified in order to solve model 2, as stated in section 5. For the CA, we used the same procedures and parameter settings proposed by (Başdere & Bilge, 2014). The experiments in this section are carried out by using all cases presented in Table 2. The summary of results obtained from the three methods can be seen in Table 6, including the same statistics as in previous sections.

Table 5: Performance characteristics of CPLEX, CA, and proposed algorithm when solving model 2.

Test Cases	CPLEX		Compressed Annealing (CA)				The proposed Algorithm				
	Z^*	$CPU(s)$	Z_{best}	$\bar{Z}$	σ_z	$\overline{CPU(s)}$	Z_{best}	$\bar{Z}$	σ_z	$\overline{CPU(s)}$	$IMP_{CA}(\%)$
Case 1	16,157	0.96	16,157	16,157	0	0.87	16,157	16,157	0	0.22	0
Case 2	2,258	2.85	2,258	2,258	0	1.93	2,258	2,258	0	0.28	0
Case 3	5,151	16.23	5,151	5,151	0	12.28	5,151	5,151	0	0.23	0
Case 4	9,655	50.08	9,655	9,655	0	31.49	9,655	9,655	0	0.23	0
Case 5	14,533	220.95	14,225	14,150	411.79	70.23	14,533	14,348	54.30	0.79	1.37
Case 6	21,337	333.07	21,141	20,796	224.48	220.47	21,337	20,950	92.87	1.44	0.73
Case 7	40,995	590.78	39,315	39,220	283.64	350.48	40,995	40,751	123.58	1.57	3.75
Case 8	32,508	8598.32	31,283	31,099	430.69	780.09	32,508	32,441	160.28	2.50	4.13
Case 9	--	--	56,210	55,320	582.34	1528.23	57,452	57,321	289.74	2.33	3.49
Case 10	--	--	67,021	65,789	865.12	1768.37	69,158	68,853	440.52	8.78	4.45

Note: $IMP_{CA}(\%) = (\bar{Z}_{proposed\ algorithm} - \bar{Z}_{CA})/\bar{Z}_{proposed\ algorithm}$

Regarding the comparison between the proposed algorithm and CPLEX, the results are almost same as the results reported in section 6.2. On the other hand, for the comparison between the proposed algorithm and CA, the proposed algorithm outperforms CA in terms of solution quality and computational time. Regarding the solution quality, both methods bring the same $\bar{Z}$ for small test instances, as in the first four cases. As the number of flight legs and aircraft increase, $\bar{Z}$ of the proposed algorithm outperforms $\bar{Z}$ of CA by about 1.37%, as in case 5, and this ratio increases gradually up to 4.45% in case 10. With respect to the computational time, the outperformance of the proposed algorithm over CA is not noticeable for the first five cases, but, for the rest of the test cases, the proposed algorithm is faster than CA. This is clearly shown in case 10 since the proposed algorithm produces the solution within, at most, 10 seconds while CA needs up to 30 minutes.

The rationale behind the outperformance of the proposed algorithm over CA lies in the following points:

- *Solution quality:* CA starts with an initial solution obtained from a simplified version of OAMR, which ignores the objective function and all the operational maintenance constraints. This results in the generation of infeasible routes with poor solution quality. In contrast to CA, the proposed algorithm starts with so-called sub-routes, as described in steps 2 and 3 of section 5. These sub-routes include all the through connects that are the source of profit maximization, so that the profit of the generated sub-routes is already maximized, leading to a good solution quality.
- *Computational time:* CA continues its procedures by concentrating on two tasks simultaneously; maximizing the profit and satisfying the operational maintenance constraints. This is very difficult, especially for large-scale test instances, because you can find a profitable route, but this route does not satisfy the operational maintenance constraints, and vice versa. This leads to the generation of a solution with poor quality in a long computational time. On the other hand, the proposed algorithm proceeds by focusing only on one task, which is connecting the generated sub-routes in such a way that the operational maintenance constraints are satisfied. This shortens the computational time significantly, and the profit of the obtained routes is already maximized.
- *Search mechanism:* CA looks for a neighbourhood solution by using a swapping technique that considers only the connection feasibility (time and place issues). This technique is not efficient for two reasons. Firstly, it easily breaks the through connects, resulting in a profit minimization of the generated solution. Secondly, cutting two strings of the aircraft routes and swapping their tails can easily generate maintenance infeasible routes. Our proposed algorithm avoids the first

drawback by building sub-routes of through connects and makes them fixed, so it is rare to break the through connects and minimize the profit. Also, our proposed algorithm avoids the second drawback by using the forward and backward insertion approaches, as described in steps 6 of section 5. The advantage of these approaches is that they look for appropriate flights or sub-routes and insert them into the complete route, while considering the maintenance constraints. So, it is rare to violate the operational maintenance constraints.

Therefore, it is clear from this section, that the proposed algorithm improves the results obtained by the existing solution methods. This echoes the importance of the proposed algorithm to be implemented in reality.

6.5 Comparison between model 1 and model 2

In this section, we report the results obtained from solving model 1 and model 2 while using the proposed algorithm. The aim of this section is to show the implication on profitability after considering the maintenance workforce capacity constraints. The statistics used in this section are $\bar{Z}_1$ and $\bar{Z}_2$ that represent the average solution obtained from model 1 and model 2, respectively. Also, the improvement ratio ($\%IMP$) is used to show the effect of considering the workforce capacity constraints. Table 7 summarizes the results obtained while solving model 1 and model 2.

Table 7: The average solutions obtained from model1 and model2.

Test cases	**Model 1** $\bar{Z}_1$	**Model 2** $\bar{Z}_2$	$\%\boldsymbol{IMP} = (\bar{\boldsymbol{Z}}_{\boldsymbol{1}} - \bar{\boldsymbol{Z}}_{\boldsymbol{2}})/\bar{\boldsymbol{Z}}_{\boldsymbol{1}}$
Case 1	16,667	16,157	3.06
Case 2	2,333	2,258	3.23
Case 3	5,333	5,151	3.41
Case 4	10,000	9,655	3.45
Case 5	14,909	14,348	3.76
Case 6	21,852	20,950	4.13
Case 7	42,542	40,751	4.21
Case 8	33,899	32,441	4.30
Case 9	59,997	57,321	4.46
Case 10	72,097	68,853	4.50

In a close look at the results in Table 7, we can see that in all cases, model 1 provides better solutions than those obtained using model 2. This point is also interpreted by the %IMP that starts from 3.06% in case 1 and increases up to 4.50% in case 10. The main reason behind this improvement is because of considering the workforce capacity, which helps planners avoid scheduling more aircraft to maintenance stations with insufficient workforce capacity. So, having no reason to ask for extra capacity, this invariably reduces the penalty cost. This situation results in reducing the total cost which in turns increases the profitability.

7. Conclusions

In this paper, we present a new MILP model for OAMRP while considering all the operational maintenance constraints along with an effective solution algorithm. In addition, the proposed model is modified to test the effect of considering the maintenance workforce capacity constraints on profitability. In terms of solution methods, first, we solve the model using commercial software (e.g. CPLEX) that can produce exact solutions for only small size test instances, with a long computational time. In order to avoid the long computational time, we propose an effective algorithm that can find high quality solutions quickly, for small and large test instances as well. In the small-scale test instances, the proposed algorithm produces best solutions that equal the exact solutions, whereas the average solutions deviate from optimality by at most 0.67%. Regarding the computational time, the proposed algorithm improves the computational time significantly, since it can find the solution within 3 seconds while CPLEX needs up to 2.5 hours to solve the problem. On the other hand, in the large-scale test instances, where CPLEX fails to even produce a feasible solution, the proposed algorithm can find the best solutions that equal the upper

bound, whereas the average solutions deviate from the upper bound by at most 0.66%. With respect to the computational time, it shows a very fast performance as the solution is found for the largest case in few seconds.

The experiments in this study are extended for two reasons. Firstly, to compare the performance of the proposed solution algorithm with existing solution methods. The results show that the proposed algorithm outperforms the existing methods like compressed annealing (CA), in terms of solution quality and computational time. Secondly, to test the effect of considering the maintenance workforce capacity. The results show that these constraints increase the profitability by 4.50% for the largest case.

There are number of future directions that can be proposed. The OAMRP in this paper is solved while the planning horizon is only 4 days. It will be interesting to solve a weekly version of OAMRP, where the number of flights and aircraft increase significantly. Also, designing an efficient solution methodology to solve the weekly version would be another research direction. In addition, the proposed OAMRP is deterministic, so proposing a stochastic model for OAMRP would be a very fruitful idea. Robustness is a pro-active way to design a flexible plan that can better withstand uncertain events. The airline industry is most likely faced by disruptions and unforeseen circumstances, so it would be a promising idea to develop robust models so as to provide appropriate solutions in such circumstances.

Acknowledgments

The work described in this paper was supported by grants from The Natural Science Foundation of China (Grant No. 71471158); The Research Committee of Hong Kong Polytechnic University (Project Numbers G-YBN1; G-YBFD; G-UA4F) and under student account code RTYN.

References

Barnhart, C., Boland, N.L., Clarke, L.W., Johnson, E.L., Nemhauser, G.L. & Shenoi, R.G. (1998). Flight String Models for Aircraft Fleeting and Routing. Transportation Science, 32(3), 208-220.

Başdere, M. & Bilge, Ü. (2014). Operational aircraft maintenance routing problem with remaining time consideration. European Journal of Operational Research, 235(1), 315-328.

Ben Ahmed, M., Zeghal Mansour, F. & Haouari, M. (2017). A two-level optimization approach for robust aircraft routing and retiming. Computers & Industrial Engineering, 112, 586-594.

Cacchiani, V. & Salazar-González, J.-J. (2016). Optimal Solutions to a Real-World Integrated Airline Scheduling Problem. Transportation Science, 51(1), 250-268.

Clarke, L., Johnson, E., Nemhauser, G. & Zhu, Z. (1997). The aircraft rotation problem. Annals of Operations Research, 69(0), 33-46.

Cohn, A.M. & Barnhart, C. (2003). Improving Crew Scheduling by Incorporating Key Maintenance Routing Decisions. Operations Research, 51(3), 387-396.

Díaz-Ramírez, J., Huertas, J.I. & Trigos, F. (2014). Aircraft maintenance, routing, and crew scheduling planning for airlines with a single fleet and a single maintenance and crew base. Computers & Industrial Engineering, 75, 68-78.

Dong, Z., Chuhang, Y. & Lau, H.Y.K.H. (2016). An integrated flight scheduling and fleet assignment method based on a discrete choice model. Computers & Industrial Engineering, 98, 195-210.

Dunbar, M., Froyland, G. & Wu, C.-L. (2014). An integrated scenario-based approach for robust aircraft routing, crew pairing and re-timing. Computers & Operations Research, 45(0), 68-86.

Eltoukhy, A.E.E., Chan, F.T.S., Chung, S.H., Niu, B. andWang, X.P. (2017a). "Heuristic approaches for operational aircraft maintenance routing problem with maximum flying hours and man-power availability considerations". Industrial Management and Data Systems, Vol.117 No. 10, PP. 2142-2170.

Eltoukhy, A.E.E., Chan, F.T.S. & Chung, S.H. (2017b). Airline schedule planning: A review and future directions. Industrial Management & Data Systems, 117(6), 1201-1243.

Feo, T.A. & Bard, J.F. (1989). Flight Scheduling and Maintenance Base Planning. Management Science, 35(12), 1415-1432.

Gopalan, R. & Talluri, K.T. (1998). The Aircraft Maintenance Routing Problem. Operations Research, 46(2), 260-271.

Gürkan, H., Gürel, S. & Aktürk, M.S. (2016). An integrated approach for airline scheduling, aircraft fleeting and routing with cruise speed control. Transportation Research Part C: Emerging Technologies, 68, 38-57.

Haouari, M., Shao, S. & Sherali, H.D. (2012). A Lifted Compact Formulation for the Daily Aircraft Maintenance Routing Problem. Transportation Science, 47(4), 508-525.

Haouari, M., Sherali, H., Mansour, F. & Aissaoui, N. (2011). Exact approaches for integrated aircraft fleeting and routing at TunisAir. Computational Optimization & Applications, 49(2), 213-239.

Hu, Y., Xu, B., Bard, J.F., Chi, H. & Gao, M.g. (2015). Optimization of multi-fleet aircraft routing considering passenger transiting under airline disruption. Computers & Industrial Engineering, 80, 132-144.

IATA. (2014). Airlines Expect 31% Rise in Passenger Demand by 2017. <http://www.iata.org/pressroom/pr/pages/2013-12-10-01.aspx>.

ICAO. (2014). 2013 ICAO Air Transport Results confirm robust Passenger Demand, sluggish Cargo Market. <http://www.icao.int/Newsroom/Pages/2013-ICAOAIR- TRANSPORT-RESULTS-CONFIRM-ROBUST-PASSENGER-DEMAND,-SLUGGISH-CARGO-MARKET.aspx>.

Kabbani, N.M. & Patty, B.W. (1992). Aircraft routing at American airlines. In: In Proceedings of the 32nd annual symposium of AGIFORS. Budapest, Hungary.

Klabjan, D., Johnson, E.L., Nemhauser, G.L., Gelman, E. & Ramaswamy, S. (2002). Airline Crew Scheduling with Time Windows and Plane-Count Constraints. Transportation Science, 36(3), 337-348.

Liang, Z. & Chaovalitwongse, W.A. (2009). The Aircraft Maintenance Routing Problem. In: W. Chaovalitwongse, K.C. Furman & P.M. Pardalos, Optimization and Logistics Challenges in the Enterprise (pp. 327-348). Boston, MA: Springer US.

Liang, Z., Chaovalitwongse, W.A., Huang, H.C. & Johnson, E.L. (2011). On a New Rotation Tour Network Model for Aircraft Maintenance Routing Problem. Transportation Science, 45(1), 109-120.

Mercier, A., Cordeau, J.-F. & Soumis, F. (2005). A computational study of Benders decomposition for the integrated aircraft routing and crew scheduling problem. Computers & Operations Research, 32(6), 1451-1476.

Mercier, A. & Soumis, F. (2007). An integrated aircraft routing, crew scheduling and flight retiming model. Computers & Operations Research, 34(8), 2251-2265.

Salazar-González, J.-J. (2014). Approaches to solve the fleet-assignment, aircraft-routing, crew-pairing and crew-rostering problems of a regional carrier. Omega, 43(0), 71-82.

Sarac, A., Batta, R. & Rump, C.M. (2006). A branch-and-price approach for operational aircraft maintenance routing. European Journal of Operational Research, 175(3), 1850-1869.

Sherali, H.D., Bae, K.-H. & Haouari, M. (2013). An Integrated Approach for Airline Flight Selection and Timing, Fleet Assignment, and Aircraft Routing. Transportation Science, 47(4), 455-476.

Sriram, C. & Haghani, A. (2003). An optimization model for aircraft maintenance scheduling and re-assignment. Transportation Research Part A: Policy and Practice, 37(1), 29-48.

Stoller, G. (2010). Planes with maintenance problems have flown anyway. In: USA TODAY.

Talluri, K.T. (1998). The Four-Day Aircraft Maintenance Routing Problem. Transportation Science, 32(1), 43-53.

Weide, O., Ryan, D. & Ehrgott, M. (2010). An iterative approach to robust and integrated aircraft routing and crew scheduling. Computers & Operations Research, 37(5), 833-844.

Zeghal, F.M., Haouari, M., Sherali, H.D. & Aissaoui, N. (2011). Flexible aircraft fleeting and routing at "Tunis Air". The Journal of the Operational Research Society, 62(2), 368-380.

Zhang, D., Henry Lau, H.Y.K. & Yu, C. (2015). A two stage heuristic algorithm for the integrated aircraft and crew schedule recovery problems. Computers & Industrial Engineering, 87, 436-453.