The following publication Eltoukhy, A. E. E., Chan, F. T. S., Chung, S. H., Niu, B., & Wang, X. P. (2017). Heuristic approaches for operational aircraft maintenance routing problem with maximum flying hours and man-power availability considerations. Industrial Management and Data Systems, 117(10), 2142–2170 is published by Emerald and is available at https://doi.org/10.1108/IMDS-11-2016-0475.

Heuristic approaches for operational aircraft maintenance routing problem with maximum flying hours and man-power availability considerations

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

[a]*Department of Industrial and Systems Engineering, The Hong Kong Polytechnic University, Hung Hum, Hong Kong*
[b]*College of Management, Shenzhen University, Shenzhen, China*
[c]*Institute of Systems Engineering, Dalian University of Technology, Dalian, China*

*Corresponding Author
elsayed.abdelrahman@connect.polyu.hk
f.chan@polyu.edu.hk
mfnick@polyu.edu.hk
drniuben@gmail.com
wxp@dlut.edu.cn

Structured Abstract:

Purpose – The purpose of this research work is twofold. Firstly, to propose an operational model for Aircraft Maintenance Routing Problem (AMRP) rather than tactical models that are commonly used in the literature. Secondly, to develop a fast and responsive solution method in order to cope with the frequent changes experienced in the airline industry.

Design/methodology/approach –Two important operational considerations were considered, simultaneously. First one is the maximum flying hours, and second one is the man-power availability. On the other hand, Ant Colony Optimization (ACO), Simulated Annealing (SA), and Genetic algorithm (GA) approaches were proposed to solve the model, and the upper bound was calculated to be the criteria to assess the performance of each meta-heuristic. After attempting to solve the model by these meta-heuristics, the authors noticed further improvement chances in terms of solution quality and computational time. Therefore, a new solution algorithm was proposed, and its performance was validated based on 12 real data from the EgyptAir Carrier. Also, the model and experiments were extended to test the effect of the operational considerations on the profit.

Findings – The computational results showed that the proposed solution algorithm outperforms other meta-heuristics in finding a better solution in much less time, whereas the operational considerations improve the profitability of the existing model.

Research limitations/implications – The authors focused on some operational considerations rather than tactical considerations that are commonly used in the literature. One advantage of this is that it improves the profitability of the existing models. On the other hand, identifying future research opportunities should help academic researchers to develop new models and improve the performance of to the existing models.

Practical implications – The experiments results showed that the proposed model and solution methods are scalable and can thus be adopted by the airline industry at large.

Originality/value – In the literature, AMRP models were cast with approximated assumption regarding the maintenance issue, while neglecting the man-power availability consideration. However, in this paper, the authors attempted to relax that maintenance assumption, and consider the man-power availability constraints. Since the result showed that these considerations improve the profitability by 5.63% in the largest case. The proposed operational considerations are hence significant. Also, the authors utilized ACO, SA, and GA to solve the model for the first time, and developed a new solution algorithm. The value and significance of the new algorithm appeared as follow. Firstly, the solution quality was improved since the average improvement ratio over ACO, SA, and GA goes up to 8.30%, 4.45%, and 4.00% respectively. Secondly, the computational time was significantly improved since it doesn't go beyond 3 seconds in all the 12 real cases, which is considered much lesser compared to ACO, SA, and GA.

Keywords:

Aircraft maintenance routing problem, Ant Colony Optimization, Simulated Annealing, Genetic algorithm.

Article Classification:

Research paper

1. Introduction

The aviation industry is now faced with the challenge of achieving a high profit margin with the incessant increase in oil price, labor, and capital. In this regard, (Liang and Chaovalitwongse, 2012) stated that U.S passenger airlines lost around $3.4 billion in 2009. Such unpleasant situations have propelled airline companies towards solving the scheduling problem using limited resources. The airline scheduling problem can be grouped into four stages namely: the flight scheduling problem (FSP), the fleet assignment problem (FAP), the aircraft maintenance routing problem (AMRP), and the crew scheduling problem (CSP). FSP deals with the construction of the flight schedule and FAP ensures that each flight leg is covered by a specific aircraft type. AMRP is the third stage and it primarily aims at designing maintenance feasible routes or a sequence of flight legs to be flown by each aircraft (tail number) with an objective of maximizing the total potential profit. In this work, we focus on AMRP. The CSP is the last stage and it solves the problem of assigning crew members to each aircraft. Although, many research efforts have been exerted to solving these problems, the challenges still exist due to the fact that the industry size is continually growing.

These challenges have motivated academic researchers and practitioners to exert more effort in improving airline scheduling. In the literature, AMRP is one of the airline scheduling problems that are widely studied with two main focuses: tactical and operational. The tactical side focuses on finding the rotations to be repeated by each aircraft, while ignoring the aircraft conditions and any sudden issues arising. The operational aspect, on the other hand, considers all aircraft conditions, since it determines the routes to be flown in real life.

Focusing on the tactical level of AMRP, (Kabbani and Patty, 1992) proposed a set partition model to find feasible routes or lines of flight (LOF). Also, (Clarke et al., 1997) developed a model that aimed at finding feasible maintenance rotations that maximize the through value. The 3-day maintenance routing problem was studied by (Gopalan and Talluri, 1998) who considered the transit and balance checks, whereas, the 4-day maintenance routing problem was tackled by (Talluri, 1998) who considered the transit check. More recently, (Liang et al., 2011) proposed a new rotation-tour time-space network for

AMRP, and solved the model using commercial software. The authors neglected the flight assignment to each individual aircraft and assumed that the maintenance is performed overnight. (Liang and Chaovalitwongse, 2012) expanded their previous work by constructing a compact representation for weekly AMRP with the assumption that the flight schedule is repeated every day of the week.

Based on operational level of AMRP, (Sriram and Haghani, 2003) were among the first authors to consider both type A and type B maintenance checks. The authors presented an effective heuristic that solved the problem in a reasonable computational time when compared with CPLEX. Also, (Sarac et al., 2006) incorporated some important operational aspects into their model that were ignored in other models, which are the resource availability constraints (human resources and maintenance stations). Moreover, (Başdere and Bilge, 2014) developed a model that considered stochasticity and the possibility of flight cancellations and delays.

All the aforementioned operational studies assumed for simplicity that each aircraft should undergo a type A maintenance check once every four days. This constraint is considered an approximation to performing the maintenance every 65 flight-hours, which is the maximum flying hours, as mandated by the Federal Aviation Administration (FAA). The consequence of this approximation is that aircraft may go for maintenance before reaching 65 flight-hour, which eventually leads to increased maintenance cost and under-utilization of flying time. So, relaxing this approximation invariably reduces the total cost and improves the financial situation of the airline. Therefore, this situation motivates us to consider the maximum flying hour constraints in our proposed model. In reality, the airline operators utilize a different strategy, in which, each aircraft undergoes type A maintenance every 35-40 flight-hours. Since our model is tailored towards real world implementation, we use 40 flight-hours to be the maximum flying hours as opposed the 65 flight-hours.

On the other hand, if, for instance, the model schedules four aircraft for maintenance in one station with insufficient man-power, it will be highly probable to face two situations. First, the wait time of some of the aircraft will be prolonged in order to receive the maintenance. Second, the wait time can be avoided if more hands and/or resources are deployed to handle the excess traffic. In both cases, additional cost will be incurred for not considering the availability of man-power. Therefore, the viability of this constraint necessitates its addition to the model proposed in this work.

In this paper, our main goal is twofold. Firstly, to develop an operational AMRP model with real considerations to help reduce the total cost incurred by airline companies, and secondly to develop a fast and responsive solution method. For this purpose, we propose solving the model using ACO, SA, and GA due to their efficiency in solving the routing problems and other different NP-hard problems. One of the obvious question after using meta-heuristics is how close to optimality the obtained solutions are. Ideally, we would like to compare the obtained solutions with the optimal solution, but sometimes it is difficult to get the optimal solution that is what motivates us to use meta-heuristics at the beginning. To compromise this situation, we propose estimating the optimistic upper bound of our objective function to be the criteria to assess the performance of each proposed meta-heuristics. In this paper, the optimistic upper bound was calculated using the greedy algorithm, since it is one of the good tools to calculate the optimistic upper bound (Zhou et al., 2015). Despite the major advantages of the selected meta-heuristics, we noticed two main drawbacks after the implementation of ACO, SA, and GA. Firstly, in the large scale test instances, the randomness during flight leg selection process sometimes destroys the through connects, resulting in a loss of their potential profit, which appeared by the gap between the average solution and the upper bound. Secondly, the computational time for these heuristics when solving large cases is quite long, which is not consistent with our aim at the beginning. This situation encourages us to develop another algorithm to improve both the solution quality and the computational time. The

performance of the developed algorithm is validated by comparing the output using ACO, SA, and GA, and by calculating the gap between the average solution and upper bound. Furthermore, the model is extended to capture the implications on profit after adding the maximum flying hour and man-power availability constraints.

The novelty in this paper is highlighted as follows:

- We propose a model without any approximation assumption regarding the maintenance issue, and without neglecting the man-power availability consideration.
- We utilize ACO, SA, and GA to solve the proposed model for the first time. These approaches yield solutions with good profitability in a reasonable computational time.
- We develop a new algorithm that outperforms ACO, SA, and GA in finding a solution with better profitability in a shorter computational time.
- We extend the model to consider the maintenance and penalty cost for the purpose of quantifying profit implications that are due to the model considerations.
- We carry out experiments using data obtained from the EgyptAir carrier, which is evidence that the proposed solution is scalable and can thus be adopted by the airline industry at large.

The rest of the paper is organized as follows. Section 2 presents a description of the model including the objective function and the constraints. The solution approaches using ACO, SA, and GA and the proposed solution algorithm are described in section 3. In section 4, the implications of considering the maximum flying hour and the availability of the man-power are presented. Section 5 covers the computational experiments and the comparison of the solution approaches. Finally, some concluding remarks and future research directions are presented in section 6.

2. The model

The proposed model of AMRP focuses on arranging maintenance feasible routes or the sequence of flight legs to be flown by each aircraft (tail number) with the objective of maximizing the total potential profit. In order to organize the maintenance feasible route, each aircraft has to undergo a type A maintenance check every 65 flight-hours or with an approximation of once every four days, as mandated by the FAA. This maintenance check involves inspection of major parts, such as the aircraft engine. Usually, the airline performs maintenance checks under different regulations, which are more stringent than those imposed by the FAA. For instance, each aircraft has to go a type A maintenance check every 35-40 flight-hours.

Since our aim in this paper is to move towards real life, we consider two important operational constraints, simultaneously. First, by performing a type A maintenance check every 35-40 flight-hours instead of 65 flight-hours or once every four days. Second, by considering the man-power availability constraints for each maintenance station, to avoid receiving aircraft that require more than the available man-power capacity.

In our proposed model, our objective is to maximize the through value, which is the revenue that comes from the additional passengers who are attracted by the through connects. Through connects occur when the connecting time between two consecutive flights covered by the same aircraft is more than the turn-around time and less than predetermined threshold determined by the airline operators, as described by (Clarke et al., 1997, Liang et al., 2011).

2.1 The model formulation (Objective function and constraints)

In this section, we present the model formulation that is based on the connection network, which is commonly used network for AMRP (Gopalan and Talluri, 1998, Haouari et al., 2012). Nodes of the network represent the flight legs, whereas, the arcs represent the possible connections between those flight legs. There are two types of the arcs in the network. First type is called ordinary arcs that connect two flight legs, whereas, the second type is called maintenance arcs that connect two flight legs such that the destination of the second flight leg has a maintenance station. To formalize the representation of the proposed AMRP, we first define the notations that are frequently used throughout this chapter, before giving the detailed formulation.

Sets

NF: the set of flight legs, indexed by i, j.
K: the set of aircraft, indexed by k.
MT: the set of maintenance stations, indexed by m.
NFM: the set of flight legs that their destination offers a maintenance check.
A: the set of airports, indexed by a.
o: the dummy source node of the network.
t: the dummy sink node of the network.

Parameters

DT_i: the departure time of flight leg i.
AT_i: the arrival time of flight leg i.
TRT: the turn-around time, which is consumed for getting passenger off, unloading the luggage, changing the gate, boarding, loading the luggage, and fuel the aircraft.
O_{ia}: the 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}: the 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: the flight duration of flight leg i.
v_{ij}: the through value of the connection between flight legs i and j.
T_{max}: the maximum flying hours between two consecutive maintenance checks.
MP_m: the man-power group that available in maintenance station m.
ET_m: the close time for the maintenance station m.
MT: the time required to perform the maintenance check
$RTAM_k$: the ready time for aircraft k to cover the next flight leg after finishing the maintenance check. Usually, it is equal the starting time for maintenance plus the maintenance time.
M: a considerable big number.
MC_k: the maintenance cost paid for maintaining aircraft k.
α_k: the aircraft delay binary indicator such that $\alpha_k = 1$ if the aircraft has a delay in the maintenance station because of unavailability of the man-power groups, and 0 otherwise.
PC_k: the penalty cost paid if the aircraft k has a time delay in the maintenance station.

Decision variables

$$x_{ijk} = \begin{cases} 1 & if\ the\ flight\ legs\ i\ and\ j\ are\ covered\ by\ aircraft\ k \\ 0 & otherwise \end{cases}$$

$$y_{ijk} = \begin{cases} 1 & if\ aircraft\ k\ covers\ flight\ leg\ i\ and\ j\ ,then\ undergo\ maintenance \\ 0 & otherwise \end{cases}$$

Based on the above notations, the AMRP formulation can be written as follows.

$$\max\ Z = \sum_{k\in K}\sum_{i\in NF}\left(\sum_{j\in NF} v_{ij}x_{ijk} + \sum_{j\in NFM} v_{ij}y_{ijk}\right) \tag{1}$$

s.t.

$$\sum_{k\in k}\left(\sum_{j\in NF} x_{ijk} + \sum_{j\in NFM} y_{ijk}\right) = 1 \qquad \forall\, i \in NF \tag{2}$$

$$\sum_{j\in NF} x_{ojk} + \sum_{j\in NFM} y_{ojk} = 1 \qquad \forall\, k \in k \tag{3}$$

$$\sum_{i\in NF} x_{itk} = 1 \qquad \forall\, k \in k \tag{4}$$

$$\sum_{j\in NF} x_{jik} = \sum_{j\in NF} x_{ijk} + \sum_{j\in NFM} y_{ijk} \qquad \forall\, i \in NF, k \in k \tag{5}$$

$$\sum_{i\in NFM} y_{jik} = \sum_{i\in NFM} x_{ijk} \qquad \forall\, j \in NF, k \in k \tag{6}$$

$$\sum_{j\in NF} x_{ojk} + \sum_{j\in NFM} y_{ojk} = \sum_{i\in NF} x_{itk} \qquad \forall\, k \in k \tag{7}$$

$$AT_i + TRT - DT_j \le M(1 - x_{ijk}) \qquad \forall\, i,j \in NF, k \in k \tag{8}$$

$$\sum_{k\in k} x_{ijk} \le \sum_{a\in A} D_{ia}O_{ja} \qquad \forall\, i,j \in NF \tag{9}$$

$$\sum_{i\in NF} FT_i\left(\sum_{j\in NF} x_{ijk} + \sum_{j\in NFM} y_{ijk}\right) \le T_{max} \qquad \forall\, k \in k \tag{10}$$

$$\sum_{i \in NF} \sum_{j \in NFM} y_{ijk} \leq MP_m \qquad \forall\, k \in k, m \in MT \tag{11}$$

$$AT_i + TRT - DT_j \leq M(1 - y_{ijk}) \qquad \forall\, i \in NF, j \in NFM, k \in k \tag{12}$$

$$AT_j + MT - ET_m \leq M(1 - y_{ijk}) \qquad \forall\, i \in NF, j \in NFM, k \in k \tag{13}$$

$$\sum_{k \in k} y_{ijk} \leq \sum_{a \in A} D_{ia} O_{ja} \qquad \forall\, i \in NF, j \in NFM \tag{14}$$

$$RTAM_k - DT_j \leq M(1 - x_{jik}) \qquad \forall\, j \in NFM, i \in NF, k \in k \tag{15}$$

$$x_{ijk} + y_{ijk} \leq 1 \qquad \forall\, i \in NF, j \in NF, k \in k \tag{16}$$

$$x_{ijk}, y_{ijk} \in (0,1) \qquad \forall\, i \in NF, j \in NF, k \in k \tag{17}$$

The objective function (1) maximizes the total potential profit that is determined by summing up the through value v_{ij} of each connection between flight legs *i* and *j*. Constraints (2), (3), and (4) constitute the cover constraints. Constraints (2) guarantee that each flight leg must be covered exactly by one aircraft. The constraints in (3) ensure that each aircraft starts its route either by using ordinary arcs or maintenance arcs. Similar to (3), constraints (4) ensure that each aircraft completes its route.

In order to maintain the circulation of aircraft throughout the network, balance constraints (5), (6), and (7) are cast. Constraints (5) indicate that if an aircraft covers flight leg by ordinary arc, then the next flight leg can be covered by using either ordinary arc or maintenance arc. Constraints (6) indicate that if an aircraft covers one flight leg by maintenance arc, then the flight leg should be covered with ordinary arc. Equal number of aircraft that start and end their routes is assured by constraints (7).

For covering two flight leg by the same aircraft, that connection should be feasible in terms of time and place. These considerations are described by constraints (8) and (9), especially for ordinary arcs. Constraints (8) indicate the time constraints such that the aircraft can cover two consecutive flight legs, if the second one departs after the arrival time of the first one plus the turn-around time. Place constraints in (9) ensure that the aircraft can cover two consecutive flight legs, if the destination of the first one and origin of the second one are the same.

It must be noted that the coverage and balance constraints do not enforce the aircraft that needs maintenance to undergo maintenance check. Therefore, the operational restrictive constraints (10) are cast. Constraints (10) describe the maximum flying hour constraints, and they limit the accumulated flight hours for each aircraft to be below T_{max}. If the accumulated flight duration exceeds the limit, the aircraft must undergo maintenance check.

In order to avoid over capacity scheduling, man-power availability constraints are considered through constraints (11). They ensure that the number of maintained aircraft does not exceed the maintenance station capacity. That capacity is represented by the number of man-power groups.

Similar to constraints (8) and (9), constraints (12), (13), and (14) indicate the time and place considerations for maintenance arcs. Constraints (12) and (13) describe the time constraints, where constraints (12) act like constraints (8), but during the usage of the maintenance arcs. Using maintenance arcs means performing maintenance check after covering the flight leg; therefore, the time constraints for

the maintenance should be cast as appeared by constraints (13), which guarantee that the aircraft will be maintained before the closing time for the maintenance station. Same as (9), constraints (14) regulate the place considerations during the usage of the maintenance arcs.
To regulate covering the flight legs after performing the maintenance, the constraints (15) are cast. They ensure that the aircraft can cover the next flight leg if its departure time is larger than or equal the aircraft ready time. Constraints (16) restrict the arc selection by the aircraft. In other word, the aircraft should cover the flight legs by using only one arc type, which be either ordinary arc or maintenance arc. Constraints (17) are the integrality constraints on variables.
The scope of the proposed AMRP model is described as follow:

- The planning horizon is 4 days.
- The model only considers the existing maintenance stations and there is no recommendation for constructing new stations.
- The maintenance stations are located in the hub airports.
- The number of man-power groups in each maintenance station is deterministic.
- The model considers only the type A maintenance check, and it is carried out only during the night.

3 Solution Methods

Before discussing different solution methods used to solve the proposed model, it is very important to mention that AMRP belongs to the class of NP-hard problems (Sarac et al., 2006). So, it is practical and reasonable to use meta-heuristic approaches, especially for large scale problems. This decision is confirmed since many researchers in related fields have successively applied meta-heuristics for solving different optimization problems. For example, travelling salesman problem (Wu et al., 2009), crew scheduling (Ozdemir and Mohan, 2001), vehicle routing problem (Wang et al., 2015), aircrew rostering problem (Lučic and Teodorovic, 1999), and control attitude behavior problem (Hashim and Abido, 2015, Hashim et al., 2015).

On the other hand, the airline industry is characterized by frequent external changes such as bad weather and technical problems, among other factors. In the case of external changes occurrence, it may lead to the infeasibility of some planned routes, which requires solving the model very frequently in order to respond to these changes. So, we need fast and responsive solution methods to solve the AMRP frequently. For this purpose, we decide using meta-heuristic approaches to solve AMRP, which fits our essential needs.

Usually, in the literature, the commonly used meta-heuristics for solving the routing problem can be broadly divided into three categories (Cordeau et al., 2007):

- learning mechanisms, including neural networks and ant colony optimization (ACO);
- local search, including simulated annealing (SA), deterministic annealing, and tabu search;
- population search, including genetic algorithm (GA) and adaptive memory procedures.

To select the best methods to solve our proposed model, a comprehensive survey was conducted. That survey resulted in selecting the best and most recommended methods to solve the routing problem. The selection includes proposing ACO, SA, and GA, because of their efficiency in solving the routing problems and other different NP-hard problems. That efficient performance of ACO, SA, and GA are shown by (Balseiro et al., 2011, Yu and Yang, 2011, Deng and Lin, 2011, Wu et al., 2009), and by (Lučic

and Teodorovic, 1999, Baños et al., 2013, Wang et al., 2015, Mak and Boland, 2000), and by (Cheng and Wang, 2009, Ghoseiri and Ghannadpour, 2010, Ozdemir and Mohan, 2001, Souai and Teghem, 2009, Yuan et al., 2013, Groba et al., 2015) , respectively.

3.1 Ant colony optimization approach for AMRP

The Ant colony optimization (ACO) approach is a meta-heuristic that appeared in the Ph.D. dissertation by Dorigo in 1992. Such an approach mimics the foraging behavior of real ants when they search for the food. Initially, each ant travels using different path and on its return trip, it deposes an amount of pheromone on its path. These pheromones act as the communication medium between the ants, and guides the other ants in the next travels.

Since there are many different versions of ACO, in our work we use the Ant Colony System (ACS) version because it uses the state transition rule for exploiting a good solution. In addition, ACS updates the pheromone trial locally and globally, so solution improvements can be achieved in the next iterations.

3.1.1 Route construction

Mapping ACO to the ARMP, each ant simulates an aircraft $k \in K$. Initially, each ant starts the route construction by selecting the flight leg $i \in NF$ with the earliest departure time among all the flight legs, then the next flight leg $j \in NF$ is selected based on the following state transition rule:

$$j = \begin{cases} \arg\_max_{l \in N_i^k} \left\{ [\tau_{ij}]^{\alpha} [\eta_{ij}]^{\beta} \right\} & if\ q \le q_0 \\ J & if\ q > q_0 \end{cases} \qquad (18)$$

where $N_i^k$is the set of potential flight legs that can be selected after flight leg i, by the $k\ th$ ant. The term τ_{ij} is the pheromone trial of the arc (i,j) , whereas η_{ij} is the heuristic function of the arc (i,j) that is equal to $1/(D_j - AT_i + TRT)$. The two parameters α and β represent the relative importance of the pheromone trial and the heuristic function respectively. q_0 is the exploration threshold parameter ($0 \le q_0 \le 1$) and q is a uniformly distributed random number [0~1].

Typically, the ant selects the next flight leg based on the value q. If $q \le q_0$, then ant makes an exploitation oriented selection by selecting the flight leg j in which its arc (i,j) has the best τ_{ij} and η_{ij}. On the flip side, if $q > q_0$, the ant makes an exploration oriented selection, by picking flight leg j according to the following probability rule:

$$P_{ij}^k = \frac{[\tau_{ij}]^{\alpha} [\eta_{ij}]^{\beta}}{\sum_{l \in N_i^k} [\tau_{ij}]^{\alpha} [\eta_{ij}]^{\beta}} \qquad if\ j \in N_i^k \qquad (19)$$

Once the flight leg j is selected, it will be added to the route, and the ant will continue selecting the flight legs using the state transition rule. The route construction is terminated when there are no more potential flight legs to be selected or when all flight legs are covered.

3.1.2 Cumulative flying hour and maintenance consideration

During route construction, ACO has to check the cumulative flying hour for the aircraft (CFH_k) to see if it does not exceed T_{max}, so that the ant can make the next move, otherwise, the ant should visit the maintenance station observing the constraints in Equations (11), (12), (13), and (14). The ant continues the route construction after performing maintenance while satisfying the constraint in (15).

3.1.3 Pheromone trial updating

This step is considered the core part of ACO, where the pheromone trial is updated to reflect the quality of the solution obtained by the ants. There are two types of pheromone trail update: locally and globally. Local pheromone trail update is carried out after each iteration, whereas global pheromone trail update is done for the best route so far.

Firstly, the local pheromone trials are updated by reducing the pheromone trial of all covered arcs in such a way to mimic the natural evaporation of the pheromones. This update process helps the ants in the next iteration to forget the old routes and scout for better routes. This can be done in accordance with following rule:

$$\tau_{ij}^{new} = (1-\rho)\tau_{ij}^{old} + \rho * \tau_0 \qquad (19)$$

where ρ is the evaporation rate parameter ($0 < \rho < 1$) and τ_0 is the initial value of the pheromone trial in all arcs (i,j) in the graph.

Secondly, the global pheromone trials are conducted to the best so far route by increasing the amount of pheromone on the arcs of that route. By making such an update, the ant will select this route in the next operation, with higher probability. Global pheromone is updated using Eq. (20).

$$\tau_{ij}^{new} = (1-\rho)\tau_{ij}^{old} + \rho * \Delta\tau_{ij}^{gb} \qquad (20)$$

where $\Delta\tau_{ij}^{gb}$ is the amount of pheromone deposited on the arc (i,j) that is included in the best route so far. $\Delta\tau_{ij}^{gb}$ is calculated using Eq. (21).

$$\Delta\tau_{ij}^{gb} = Q * v(gb) \qquad (21)$$

where Q is the control factor of laying the pheromone, in which its value determines whether to converge to the local optimal or to search randomly. The term $v(gb)$ is the sum of the through values of all arcs included in the best route so far.

The main procedure of ACO used to solve our proposed AMRP is shown in figure 1. This procedure constitutes one iteration that is repeated until the stopping criteria are satisfied.

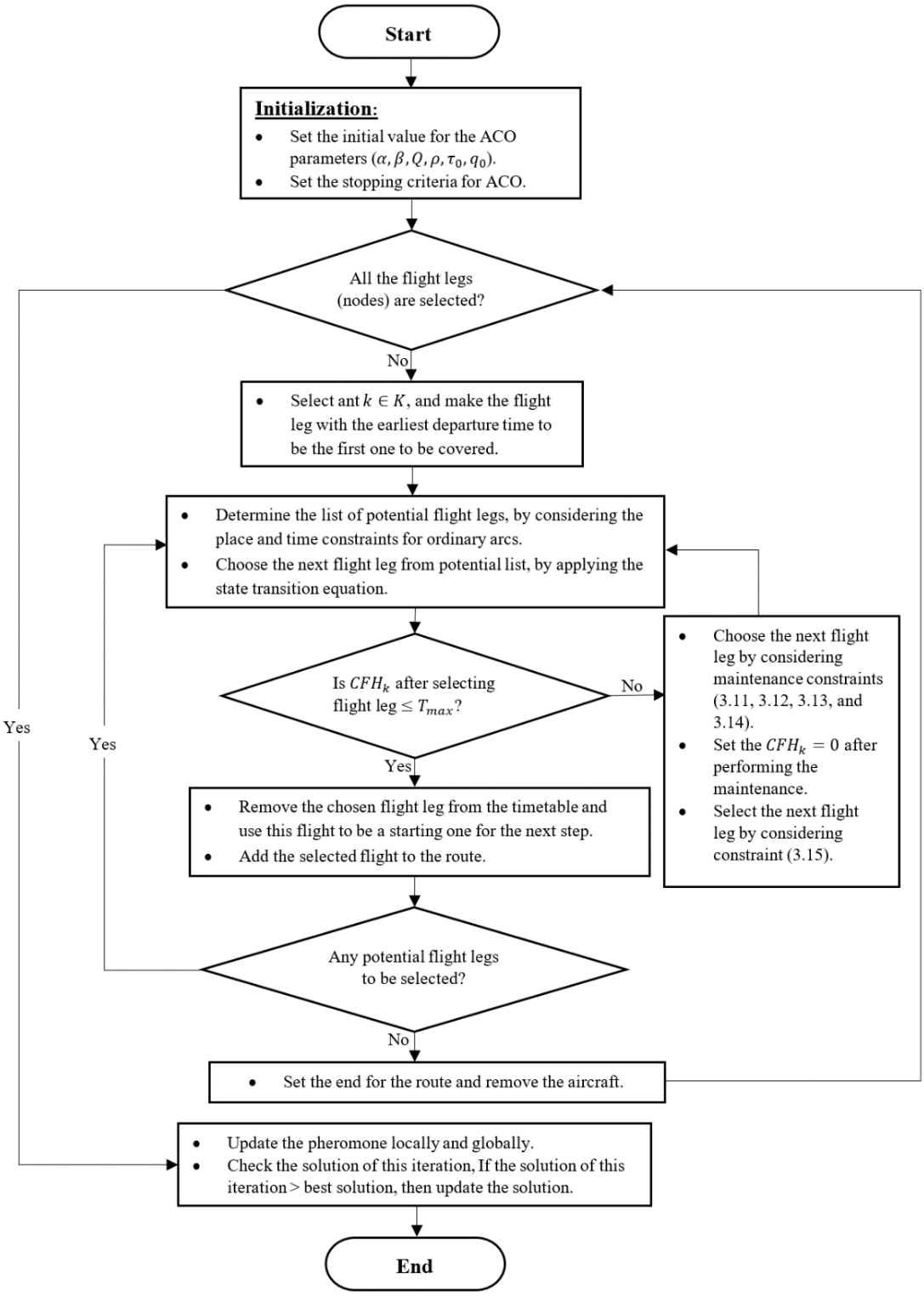

Figure 1: One iteration procedure of ACO for AMRP.

3.2 Simulated annealing approach for AMRP

The Simulated annealing (SA) is a well-known meta-heuristic that was proposed by (Kirkpatrick et al., 1983) to solve optimization problems by mimicking the process of metal annealing. SA as a search algorithm starts with an initial solution and moves in the solution space to find a better solution. During the search, if a new better solution $S`$ is identified, then $S`$ is accepted and replaces the old solution S, after which SA continuous to search based on $S`$. On the other hand, if the new solution is worse than the current best, SA can accept it with a probability that basically depends on the temperature; this is done to avoid being trapped in the local optimum. Initially when the temperature is high, SA accepts a worse solution with high probability, and when the temperature becomes low, the probability of accepting a worse solution becomes small. The details of the proposed SA approach are explained in the following sections.

3.2.1 Initial solution construction

In order to construct the initial feasible solution, we design a simple and efficient method to obtain a feasible route for each aircraft $k \in K$. The method consists of the following steps:

a. Pick an aircraft $k \in K$ from the aircraft list and assign it to cover the flight $i \in NF$, with the earliest departure time to be the starting flight in the route.
b. Determine the list of potential flight legs to be covered later by using the place and time constraints for ordinary arcs, as described in Equations (8) and (9). Then, calculate the through value for each connection.
c. Select the flight leg with the highest through value in the list to be the next flight leg covered in the route.
d. Check $CFH_k$for the aircraft after selection. If $CFH_k > T_{max}$, then go to step f, otherwise go to step e.
e. Add the selected flight leg to the route and remove it from NF. Consider the selected flight leg to be the basic for the next selection process, then go to step g.
f. If $CFH_k > T_{max}$, the next flight should be selected according to the maintenance constraints in Equations (11), (12), (13) and (14). After finishing the maintenance, set $CFH_k = 0$, select the next flight leg to be covered by the aircraft by considering the constraint in Eq. (15), then go to step g.
g. If there are any potential flights legs to be selected later, then go to step b, otherwise go to step h.
h. Set the end for the current route, remove the aircraft from the list, then go to step a.

This algorithm is terminated when all the flight legs are covered by the aircraft in the fleet.

3.2.2 Local search improvement

This step is essential in SA for finding a better solution among neighborhood solutions. In this paper, we adopt the commonly used, efficient, 2-opt move method proposed by (Potvin and Rousseau, 1995) to find a neighborhood solution. The method works as follows: for a given two routes, two arcs are removed from each route, in such a way that each route is divided into two parts as shown in figure 2. The next step is to swap the second part of each route.

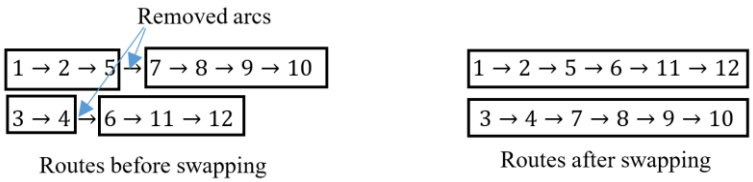

Figure 2: The 2-opt move method.

In order to avoid selecting the removed arcs randomly, which can lead to an infeasible route after swapping, we considered place and time constraints during the swapping process as enforced by the constraints given in Eq. (8) and (9). For simplicity, we do not take CFH_k into account and so we move it to the next step.

3.2.3 Checking CFH_k and Maintenance constraints

For each constructed route covered by aircraft $k \in K$, the CFH_k is checked. If CFH_k doesn't exceed T_{max}, then the constructed route for aircraft $k \in K$ is feasible, otherwise the route should be reconstructed by removing all the flight legs violating T_{max}, as shown in figure 3. After the removal, the aircraft should undergo maintenance by considering the constraints given in Eq. (11), (12), (13), and (14). Each removed flight leg should be inserted on the location that maximizes the through value, while considering place

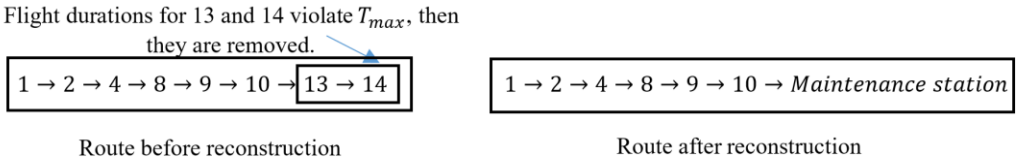

Figure 3: Route reconstruction while considering the maintenance constraints.

and time constraints.

3.2.4 Evaluating the new solution

After the local search improvement process, a new solution $S`$ with objective function value $Z`$ is obtained, whereas, the old solution S has an objective function value Z. If $Z` > Z$, SA accepts $S`$ and S is replaced by $S`$, otherwise the worse solution is accepted based on the following probability:

$$P = \frac{1}{1 + e^{-\Delta Z/T}} \qquad (22)$$

where ΔZ is the difference between new and old objective function values, and T is the temperature of the current step. By accepting the worse solution, SA avoids being trapped in a local minimum and continues exploring to find the global minimum. The temperature in our proposed SA is determined according to the following rule:

$$T_{c+1} = r * T_c \qquad (23)$$

where T_c and T_{c+1} are the temperatures of the current and next step respectively, and r is the cooling rate. The SA procedure employed in solving our proposed AMRP is shown in figure 4.

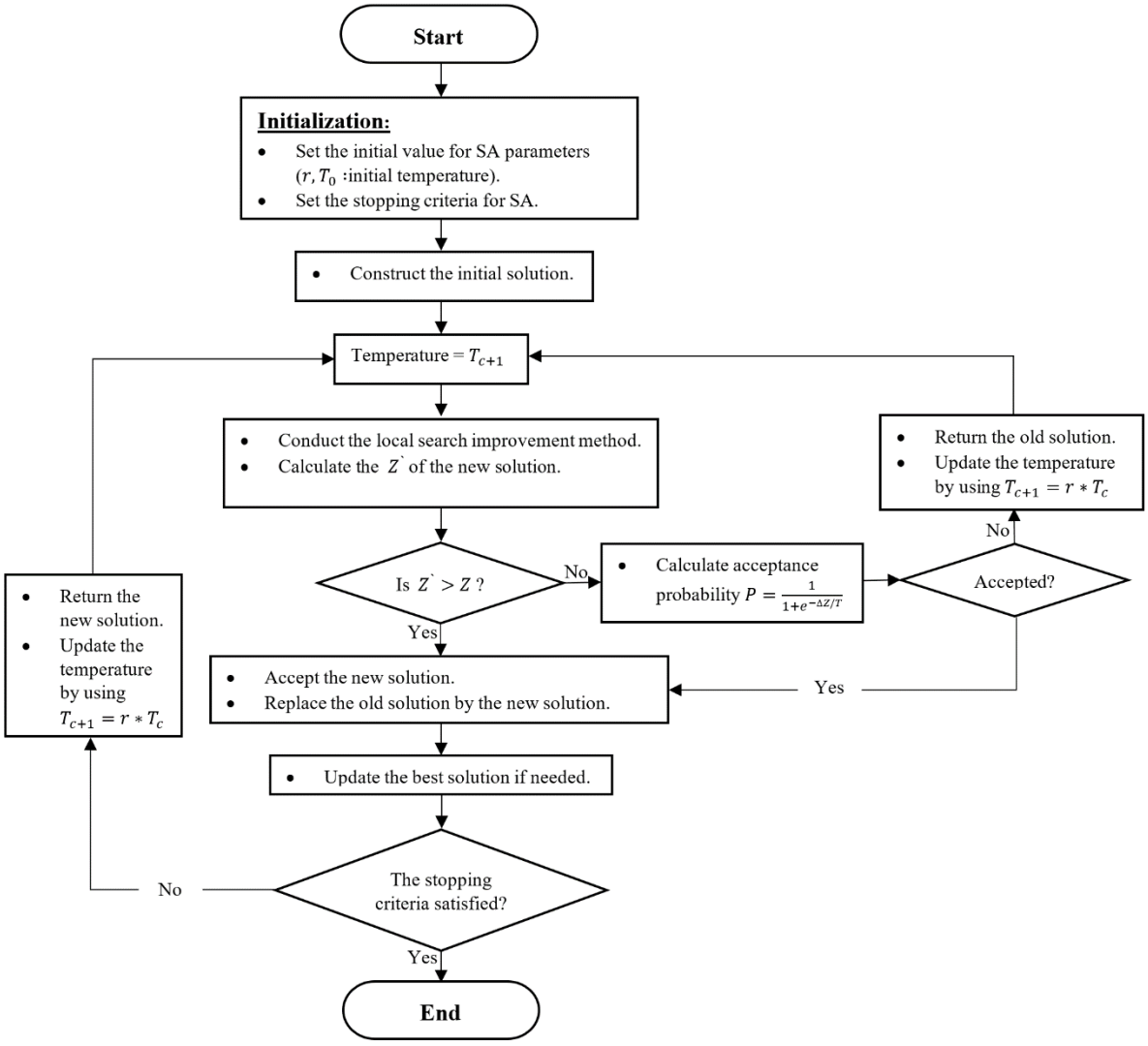

Figure 4: The procedure of SA for AMRP.

3.3 Genetic Algorithm for AMRP

The Genetic Algorithm (GA) is another well-known adaptive heuristic method that was proposed in 1975 by John Holland, at the University of Michigan. GA was proposed based on natural evolution concept of "survival of the fittest" developed by Darwin. The GA starts by constructing the initial population, where each solution of the population is represented in the form of string (chromosome). Then, the GA produces a new generation through a crossover process, while two parent chromosomes are selected for mating to produce offspring. After that, GA mutates some chromosomes from the population, in order to introduce some randomness to avoid local optimum convergence. Each iteration of GA goes through this process until the stopping criteria is satisfied. The GA we adopt is explained in the following sub-sections.

3.3.1 Chromosome representation

In GA, each solution is represented as a chromosome, which consists of two main entities, aircraft and routes. The aircraft are represented with integer numbers in a separator in front of the route, whereas, the route is represented by a chain of integer numbers. In GA, each chromosome should have a fitness value given as the value of the objective function. Figure 5 shows the typical representation of one chromosome that consists of 3 aircraft and 12 flight legs.

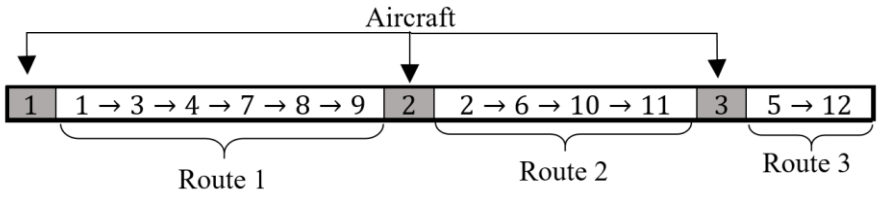

Figure 5: The representation of the chromosome.

3.3.2 Initial population construction

The initial population is constructed by using the same method explained in section 3.2.1, except for introducing some randomness in step c, in order to encourage more exploration of the search space. In step c, the next flight leg can be chosen in two different ways. The first is by selecting the flight leg that has connection with highest through value while the second way is selecting the flight leg randomly. In order to select which method to be used, we design a threshold $tr \sim [0,1]$ and generate a random number. If this random number $\leq tr$, then the first way is used, otherwise the second way is used. This algorithm is iterated until the initial population is constructed.

3.3.3 Tournament selection (TS)

TS is the process that aims to select the required chromosomes for the mating process. A set of chromosomes is chosen from the population and a tournament competition is conducted among them. The winner in the tournament (the chromosome with the highest fitness value) fills the mating pool. This competition is repeated until the mating pool is filled with the required number of chromosomes. TS provides adjustment for the selection pressure by changing the competition size. If the size is large, weak chromosomes have little chance of being selected, and vice versa.

3.3.4 Best cost-best route crossover (BCBR)

Our proposed GA adopts BCBR crossover that was developed by (Ghoseiri and Ghannadpour, 2010) due to its efficiency to produce feasible children. BSBR selects the routes based on the average cost. Since our objective is maximizing the profit, then the selection criteria is adjusted, so that the routes are selected based on the best profit instead of average cost. In BCBR crossover, two parents (chromosomes) are selected randomly from the mating pool in order to produce two children (offsprings). Each parent consists of a number of routes, and the route with highest profit is selected from each parent. Then, these routes' flights are removed from the other parent. Figure 6 shows an example of how BCBR is conducted. Suppose that route 1 and route 2 are the best profit routes in the first and second parent respectively, then the selected route flights should be removed from the other parent, as shown in figure 6. After that, each removed flight should be inserted in the location that maximizes the through value. To avoid building infeasible routes during the insertion step, place and time constraints have to be considered, as explained in the constraints given in Eq. (8) and (9). BCBR is repeated until the new generation is fully produced.

For simplicity, we do not take CFH_k and maintenance into account during the crossover process and so we consider them after building the routes, by following the same approach described in section 3.2.3.

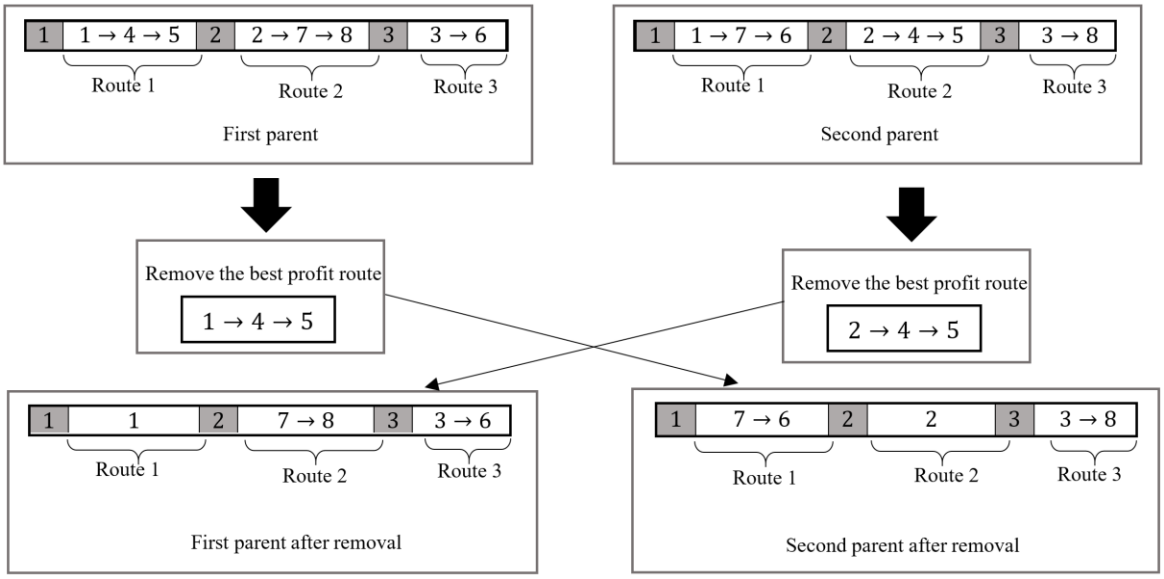

Figure 6: Example of BCBR crossover operator.

3.3.5 Sequenced based mutation (SBM)

In order to make a mutation, our GA applies SBM operator due to its efficacy and appropriateness while solving the routing model. SBM introduces some randomness to avoid local optimum convergence. It is conducted by selecting two children generated from the crossover process. Then, one arc is selected from each route as a breaking point. Each breaking point (BP_1, BP_2) breaks the route into two parts, before and after the breaking point. In order to mutate, these parts are exchanged. In a nutshell, the first new children is constructed by keeping the route part before BP_1, while the route part after BP_1 is replaced with the route part after BP_2. Similarly, the second new children is constructed by replacing the route part after BP_2 with the route part after BP_1, while keeping route part before BP_2. Details of how SBM is conducted is illustrated in figure 7.The main drawback of this mutation operator is the production of infeasible chromosomes, which leads to routing one flight leg more than once or losing other flight legs. A close look at the first new children in figure 7, we can see that flight leg 6 is duplicated, while flight legs 7 and 8 are unrouted. In order to review this situation, we remove duplicated flights and unrouted flights are inserted on the location that maximizes the through value. If, after the review process, the chromosome is still infeasible, then it will be replaced by the original children. The flowchart of the proposed GA is depicted in figure 8.

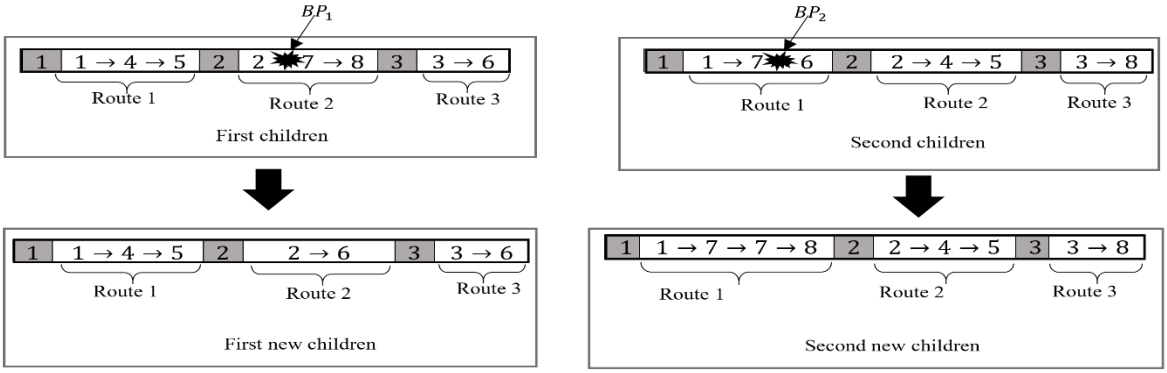

Figure 7: SBM operator for producing two new children

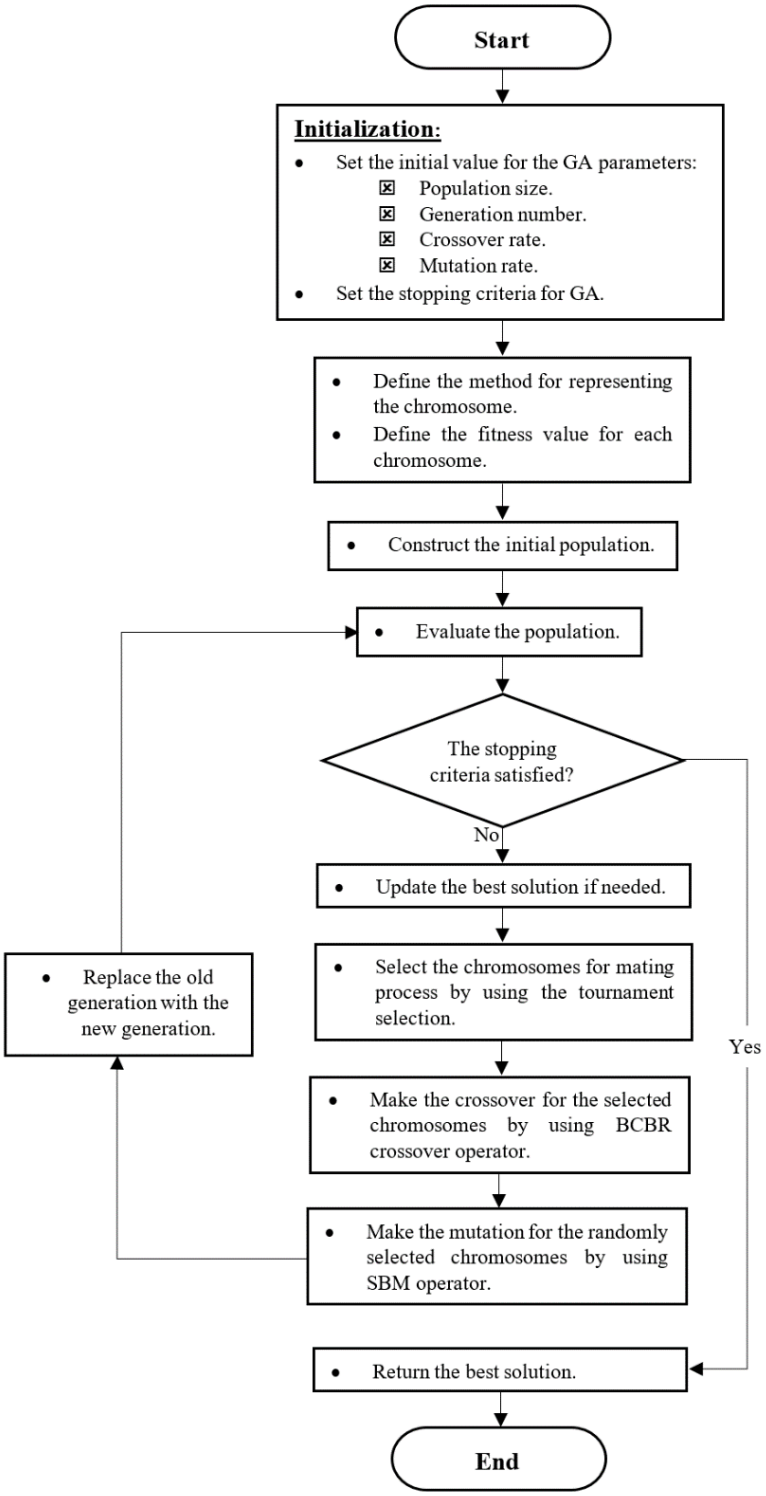

Figure 8: Flowchart of GA used for AMRP.

3.4 Evaluating the meta-heuristics performance

One of the obvious questions after using meta-heuristics is how close to optimality the obtained solutions are. Ideally, we would like to compare the obtained solutions with the optimal solution, but sometimes it is difficult to get the optimal solution that is what motivates us to use meta-heuristics at the beginning. To compromise this situation, we propose estimating the optimistic upper bound of our objective function to be the criteria to assess the performance of each proposed meta-heuristics. In this paper, the optimistic upper bound was calculated using the greedy algorithm, since it is one of the good tools to calculate the optimistic upper bound (Zhou et al., 2015). The procedure of the greedy algorithm can be described as follow:

1. Initialize upper bound (UB)=0
2. Prepare the flight leg list NF.
3. Pick flight leg $i \in NF$ and determine the list of potential flight legs to be covered later by using the place and time constraints for ordinary arcs, as described in Equations (8) and (9). Then, calculate the through value for each connection.
4. For each flight leg in the potential list, we make a check for any better through connection. If a better through connection is found, then this flight leg is removed from the potential list.
5. Select the highest through value, remove the flight leg i from NF list.
6. Updated UB.
7. Check NF. If it is not empty, then go to step 3, otherwise, go to step 8.
8. Output UB.

Despite the major advantages of the selected meta-heuristics, we noticed two main drawbacks after the implementation of ACO, SA, and GA. Firstly, in the large scale test instances, the randomness during flight leg selection process sometimes destroys the through connects, resulting in a loss of their potential profit. In other word, when flight 1 has four flights as a potential connected flights, and only one of them is a through connect. Sometime, due to randomness feature of meta-heuristics, that through connect, which is the best choice, is not selected. This problem affects the solution quality, since the difference between the average solution and UB goes up to 8%, 5%, and 4% with ACO, SA, and GA respectively. Secondly, the computational time for these heuristics when solving large cases is quite long, which is not consistent with our initial aim of this study. This situation encourages us to develop another algorithm to improve both the solution quality and the computational time.

3.5 Overview of the developed solution algorithm for AMRP

In this section, we describe the proposed algorithm for solving AMRP. The main idea in this algorithm is splitting the flight leg nodes NF into two pools; the first pool contains the through connects while the second pool contains the remaining flight legs. Then, the aircraft routes are constructed using these two pools, while considering the model constraints. Details of the algorithm are explained as follows.

3.5.1 Splitting flight leg nodes pool into two pools

For a given NF, the first step is to split this pool into two pools, in order to keep the through connect nodes far away from randomness breaking problem, which constitutes the main drawbacks of the proposed meta-heuristics. The first pool is called star pool SP, and has higher priority during route construction because it stores the star connects. The remaining flight leg nodes construct the second pool

called the non-star pool NS, which has low priority during the route construction. To construct SP and NS, each flight leg $i \in NF$ is picked, then the connecting time between flight leg i and other flight legs is calculated based on the following rule.

$$CT_{ij} = DT_j\text{-}AT_i \quad \forall\, i,j \in NF, i \neq j \qquad (24)$$

if the connecting time of the pair i and j has a non-zero through value. Then, this pair is a through connect that has to be deleted from NF and moved to SP. Otherwise, the flight leg i is deleted from NF and moved to NS. This calculation is repeated for each flight leg $i \in NF$ until all the flight legs in NF are exhausted.

3.5.2 Constructing sub-routes using star pool

The main objective of this step is to build extra shield around the through connects by joining them, if possible. Sub-routes are constructed by connecting two pairs of flight legs stored in SP, if such pairs exist. Two pairs can be connected, especially when the ending flight leg of one pair and the starting flight leg of the second pair are the same. To clarify this step, figure 9 presents a six-pair flight example. The first and second pairs can be connected, since they both share leg 3 as their ending and starting flight legs respectively. In a similar manner, the third and fifth pairs can be connected as well. The rest of the pairs, such as the fourth and last pairs, are kept the same for obvious reasons. Subsequently, the sub-routes are constructed and stored on a set called sub-route SR.

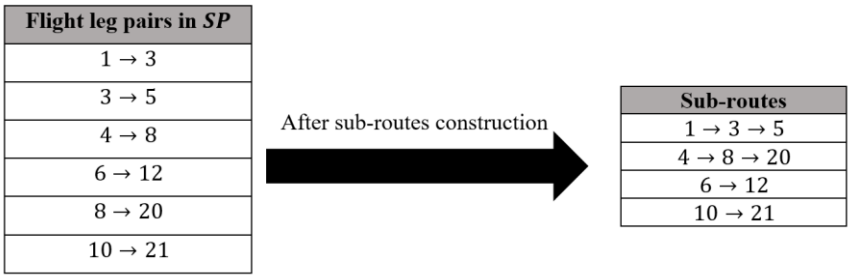

Figure 9: How sub-routes are constructed.

3.5.3 Constructing routes using sub-routes and non-star flight legs pool

The aircraft routes are constructed based on a backward and forward insertion approach using SR and NS. Since SR contains flight legs with higher priority, it is used first by randomly selecting one sub-route. If SR is empty, then the low priority set NS is used. After selecting the starting entity from SR or NS, the route can then be constructed by inserting potential sub-routes or non-star flight legs before (backward insertion), then after (forward insertion) the starting entity. At first, the backward insertion approach is applied as follows:

- For the selected entity, the starting flight leg of that entity is identified, which is either the first flight leg in the case of the sub-route or the non-star flight leg itself, in the other case.
- We search for appropriate flight leg to be inserted before the starting flight leg by firstly scanning through SR due to its high priority. If there is no potential sub-route to be selected, then the second option is used by searching through NS. The search is conducted by considering place and time constraints, as described by constraints given in Eq. (8) and (9).

- After the search, a list of potential sub-routes or non-star flight legs is identified. Then, the connecting time and the corresponding through values are calculated.
- The sub-route or non-star flight leg with the highest through value is chosen from the potential list, added to the route, and removed from SR or NS.
- The starting flight leg is updated as the first flight leg in the case of the sub-route, or it is the selected non-star flight leg itself.

This procedure is repeated until there are no more potential sub-routes or non-star flight legs to be inserted. After finishing the backward insertion process, we start inserting sub-routes and non-star flight legs using the forward insertion approach. It follows the same procedure as the backward insertion except for two points:

- Firstly, instead of identifying the starting flight leg in backward insertion, we identify the ending flight leg of the selected entity, which is either the last flight leg in the case of the sub-route or the non-star flight leg itself, in the other case.
- Secondly, the CFH_k is checked to make sure it doesn't violate T_{max} while selecting the sub-route or non-star flight leg. During the selection, if T_{max} isn't violated, then the selection is accepted. Otherwise, the forward insertion is terminated. In the case when CFH_k equals T_{max}, the selection is accepted if its ending flight leg arrives on an airport that has a maintenance station and enough man power to perform the maintenance.

To clarify the backward and forward insertion approaches, figure 10 shows a simple example of how to build the route using these approaches. At the beginning, we start selecting from the set with higher priority, which is SR. The sub-route $4 \rightarrow 8 \rightarrow 10$ is selected randomly as the starting entity. After this, the backward insertion approach is applied, and flight leg 4 is identified as the starting flight leg. Since there is no potential selection from SR, so NS is used and the flight leg 2 is the candidate flight leg to be inserted backwards. Again, the backward insertion procedure is re-applied, although there is no potential selection in both SR and NS. In this case, the backward insertion approach is terminated and the forward insertion approach is applied, where flight 10 is considered as the ending flight leg of the starting entity. In the forward insertion approach, flight leg 11 is selected from NS and the sub-route $15 \rightarrow 16$ is selected from SR. These two selections are inserted forwards, as shown in the final route constructed in figure 10.

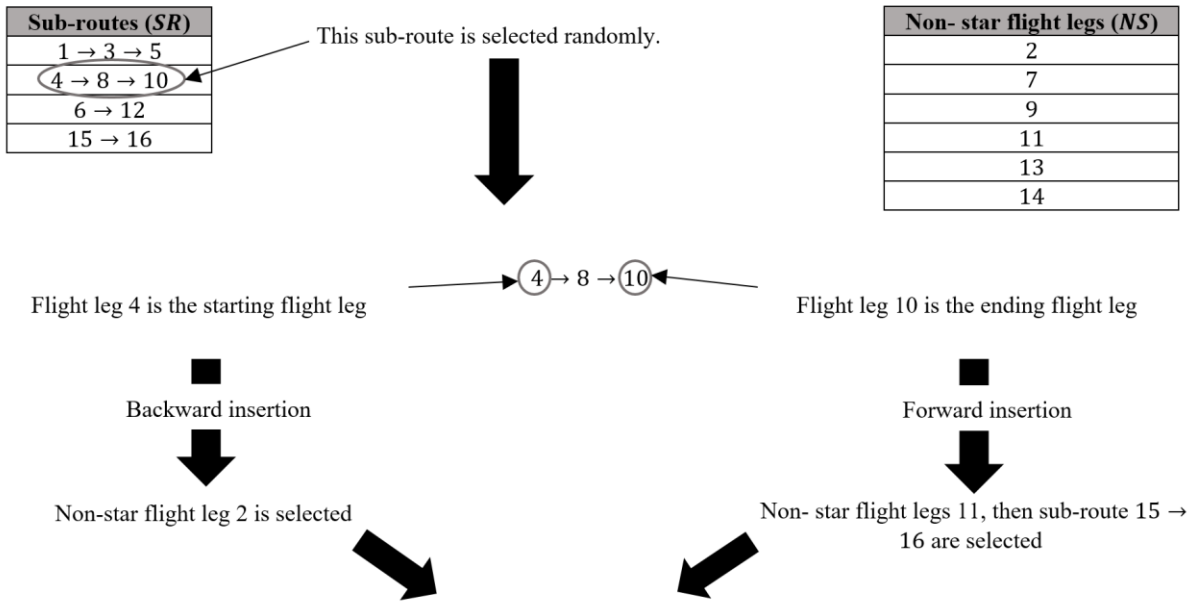

Figure 9: How backward and forward insertion approach is applied for route construction.

3.5.4 Adding the maintenance visit

After constructing the routes, this step is applied for each route that has its CFH_k equals T_{max}. Since the last flight leg of that route arrives at an airport that has a maintenance station and sufficient man power, so the maintenance can be performed directly at that airport. After the maintenance, if there are some non-star flight legs or sub-routes that are unrouted, they can be inserted by considering the constraint given in Eq. (15).

Figure 11 illustrates the flowchart of the proposed algorithm. The developed algorithm is characterized by the following features:

- it pays attention to the through connects, by identifying them and constructing a separate pool (star pool) for them as described in step 3.5.1. More attention is paid by constructing sub-routes using through connects as explained in step 3.5.2. The rationale behind this is that these two steps build a shield around the through connects; hence, any chance to break the connection between through connects is avoided. Therefore, gaining the maximum benefit from the through connects and their values can be achieved.
- During the route construction, we concentrate on the through connects by giving higher priority to SR. This concentration leads to a guarantee that all the through connects are almost routed.

These two features serve as a guide to the developed algorithm in selecting the predetermined through connects. These features, which do not exist in ACO, SA, and GA, constitute a remedy for the drawbacks appeared while using the proposed meta-heuristics. Moreover, these features result in the outperformance of the developed algorithm over the other heuristics, as seen in the preliminary results obtained from the developed algorithm.

Using such algorithm brings many benefits. For example, it will help the airline companies to construct routes with higher potential revenue, because of conducting steps 3.5.1 and 3.5.2. This will result in increasing the total profit of the airline companies. It is worth noted that these special two steps do not exist in ACO, SA, and GA. Currently, the airline companies change the aircraft routes frequently during disruptions, which make the need of quick tool for this purpose is essential. Since the efficient structure of

the algorithm will result in constructing the aircraft routes quickly, it will be appreciated by the airline industry and easily implemented.

Although the developed algorithm outperforms the other three meta-heuristics, there is one main downside for this algorithm. Actually, this algorithm is a specific algorithm, which means it is not flexible, as it is tailored especially for solving AMRP efficiently, but it fails to handle any other problem. In contrast, the other three meta-heuristics can handle different type of problems.

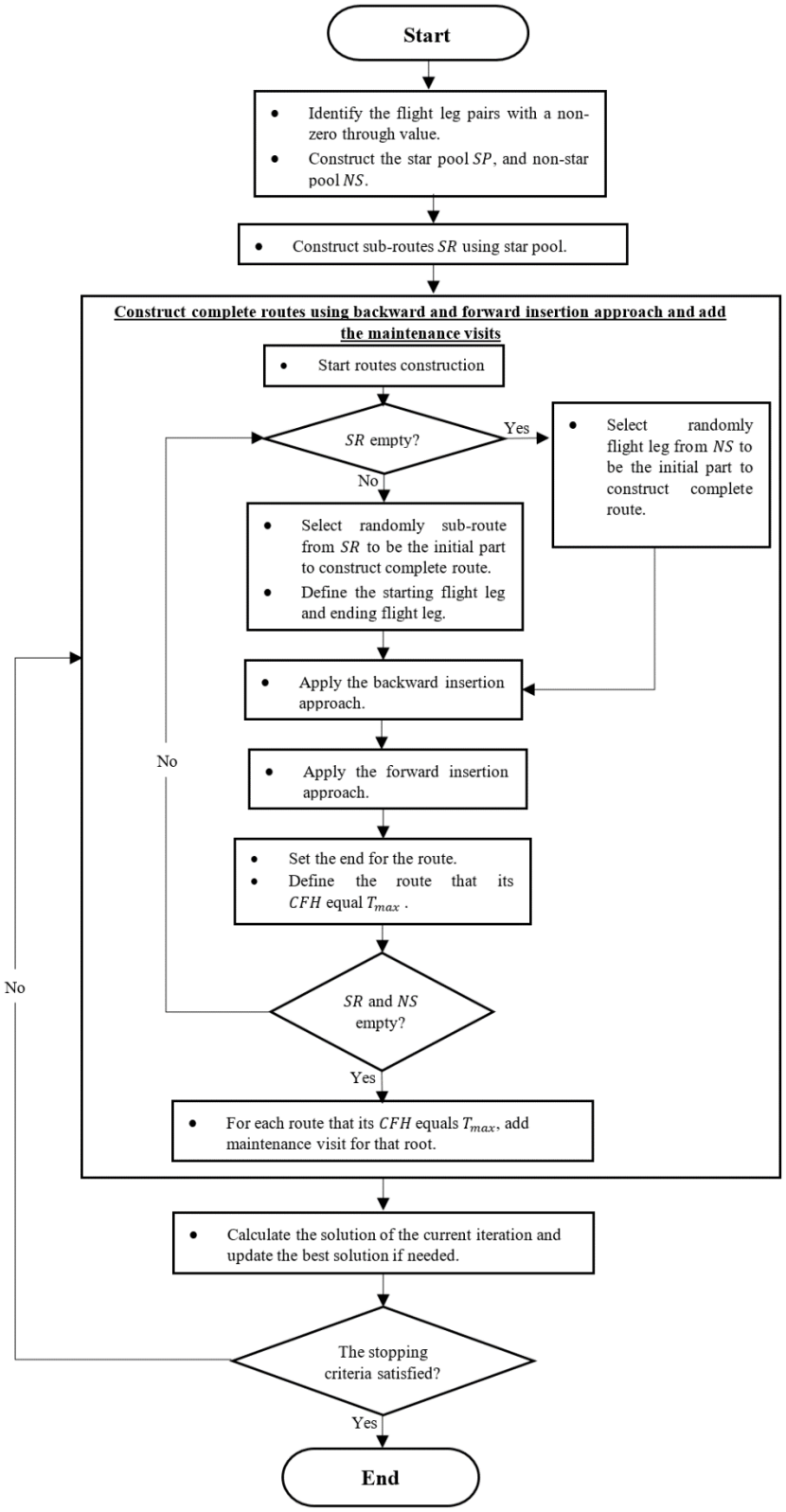

Figure 10: Flowchart of the developed algorithm.

4 Extension

Since our model considers the maximum flying hours and man-power availability constraints simultaneously, it is very important to test the implications on profitability after considering these constraints. For this purpose, two configurations of the model are presented. The first configuration, designated as "with consideration", is our proposed model which considers the constraints given in Eq. (10) and (11). The second configuration is called "without consideration ", which represents the majority of models in the literature like (Clarke et al., 1997, Liang et al., 2011), neglects man power availability constraints and considers a single visit every four days. Table 1 presents the objective function and the constraints for each of the configurations.

The objective function in this section is modified to consider the maintenance and the penalty costs, for the purpose of reflecting the impact of considering the constraints given in Eq. (10) and (11). The maintenance cost, which is the second term in the objective function, reflects the impact of considering the maximum flying hour constraints instead of considering one maintenance visit every four days. On the other hand, the penalty cost, which is the last term in the objective function, reflects the impact of considering man-power availability constraints. The penalty cost is the extra money paid when the aircraft has to wait in a maintenance station, because the number of aircraft is much more than the available man power can cope with. This situation might occur when the model ignores the man-power availability constraints, as happened in the second configuration.

From our preliminary study, the proposed solution algorithm outperforms the ACO, SA, and GA in terms of better profit and shorter computational time. These results geared us towards using the proposed algorithm to solve the two configurations presented in this section. The first configuration can be solved with the algorithm described in section 3.5. The second configuration can be solved by using the same algorithm with small modification. The only modification is to relax the T_{max} condition in section 3.5.3, consider one maintenance visit every four days, and neglect the step in section 3.5.4.

5 Computational Results

In this section, we report the results of the computational experiments conducted for assessing the performance of the proposed meat-heuristics as well as the developed algorithm. The experiments were carried out using 12 real flight schedule data sets from the EgyptAir carrier. All the test cases were carried out on an Intel i7 2.50 GHz laptop with 8 GB of RAM memory running on Windows 8 operating system. All the algorithms proposed in this paper were coded in Matlab R2014a.

5.1 Test instances

The twelve test instances used in our experiments are real schedules. In particular, our first ten cases were constructed by extracting ten flight schedules in which each schedule is covered by different fleet. In order to generate larger test cases for testing purposes, the last two cases are constructed by combining the flight schedules of multiple fleets. For example, SIM01 is constructed by combining the flight schedules of cases 7 and 9, and SIM02 is constructed by combining the flight schedules of cases 9 and 10.

For all test instances, we assume that the turn-around time TR is 45 minutes, the maximum flying hours since last maintenance T_{max} is 40 hours, and the maintenance time is 4 hours. Also, we assume 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. Detailed information about the test instances are presented in Table 2.

5.2 Parameters setting for proposed meta-heuristics

The first algorithm used to solve our model is ACO. We use the same setting applied by (Deng and Lin, 2011) who investigated different settings for α, β, and q_0. The best parameter settings reported in that study are used in our experiments, as shown on the first row of Table 3. The parameters used for SA are also presented on the second row of Table 3, which are commonly used parameters in the literature. With respect to the parameter setting for GA, we follow the same setting used by (Ghoseiri and Ghannadpour, 2010) since that parameters provide good performance, except for the population size. We set the population size to 60 instead of 100 to reduce the computational time, since there is no significant improvement using a population size greater than 60. The third row of Table 3 presents the GA parameters setting. For the proposed algorithm, the stopping criteria are described on the last row of Table 3.

Since ACO, SA, and GA are non-deterministic meta-heuristics, each experiment is replicated to measure the average performance of the experiment. Therefore, all the test cases are replicated 10 times, and the results are averaged.

5.3 Performance Characteristics of ACO, SA, and GA

Table 4 presents the solution of AMRP using the proposed heuristics. Z_{best} represents the best solution, whereas $\bar{Z}$ represents the average solution. The Gap (%) represents the difference between UB and $\bar{Z}$, and is computed by $(UB - \bar{Z})/UB$.

We can see from Table 4 that for small size data instances, as in cases 1 and 2, the Z_{best} and $\bar{Z}$ obtained from the three heuristics are equal to UB and gap is zero. By increasing the size of data instance, Z_{best} and $\bar{Z}$ of the three heuristics failed to reach the UB, as shown for cases 3 to SIM02, except for the GA that reached UB in case 4. It is also noticeable that the performance of GA and SA is almost the same in most of the cases, since their gaps are almost equal. The performance of ACO is good for small size instances like case 1 and 2, but becomes quite poor compared to other algorithms, especially for the large scale data set shown in cases 10, SIM01, and SIM02.

On the other hand, the computational time and number of iterations of stopping are presented in Table 5. The computational time are obtained from the Matlab's internal calculations function. The $CPU(s)$ records the average computational time in seconds, whereas the number of iterations until convergence is represented by the iterations for stopping.

A close look at the results in Table 5, we can see that the most computationally efficient method is SA, which can find the solution for small test instances within 3 seconds as in cases 1, 2, 3, and 4. But for large test instances, it takes much longer time. The second most computationally efficient method is ACO, which can find the solution within 1 minute for cases 1 through 9, but the CPU execution time went up to 6 minutes for the last case. It is noted that the computational time for the GA is reasonable for small test instances like cases 1 to 5, but it is quite time consuming, especially for cases 10, 11, and 12, since the flight legs and aircraft numbers increased.

5.4 Performance Characteristics of the developed solution algorithm

In order to make a caparison between the developed algorithm and the other three meta-heuristics, we select three criteria for comparison, which are commonly used in the literature. Firstly, which algorithm provides higher solution quality, as summarized through the improvement ratio in the last three columns of Table 6. In order to measure how close to optimality the obtained solutions are, the gap between upper bound and obtained solution is selected to be the second criteria for comparison between the four solution approaches. The last criteria for comparison is the computational time, as it is crucial point for airline industry.

The summary of solutions obtained from the developed algorithm can be seen in Table 6, where we report the same statistics as in Table 4 and 5. In order to comment on the improvement over the other heuristics, we evaluated the improvement ratio as shown in the last three columns of Table 6.

As seen in Table 6, Z_{best} reached UB in all the cases, whereas $\bar{Z}$ reach UB in cases 1 to 6. For the remaining cases, $\bar{Z}$ deviates slightly from UB with a gap of around 0.01, as in case SIM02. Also, it is worthy of note that the gap produced in all the cases is less than 0.1%.

Regarding the computational time, it is clear cut that the developed algorithm converges faster in all the cases. Since, the Z_{best} obtained is equal to UB, then there is no room for further improvement, so the algorithm terminates quickly. In all the cases, the developed algorithm solves the problem in less than 3 seconds.

To benchmark the performance of the developed algorithm with ACO, SA, and GA, the improvement ratio is evaluated and used as a figure of merit. The results show that, for all cases, the average improvement ratios over ACO, SA, and GA are 8.30%, 4.455, and 4.00% respectively.

Generally, we observed that the developed algorithm outperforms the other heuristics in finding a solution with better profitability in 9 out of 12 cases, as shown by boldface figures in Table 6. For more clarification, figure 12 shows the average solution obtained using the four methods. Again, the proposed algorithm succeeded in finding the solution quickly for all the cases in much less computational time compared to ACO, SA, and GA. This fast execution shows that the developed algorithm can be a potential tool for solving real AMRP.

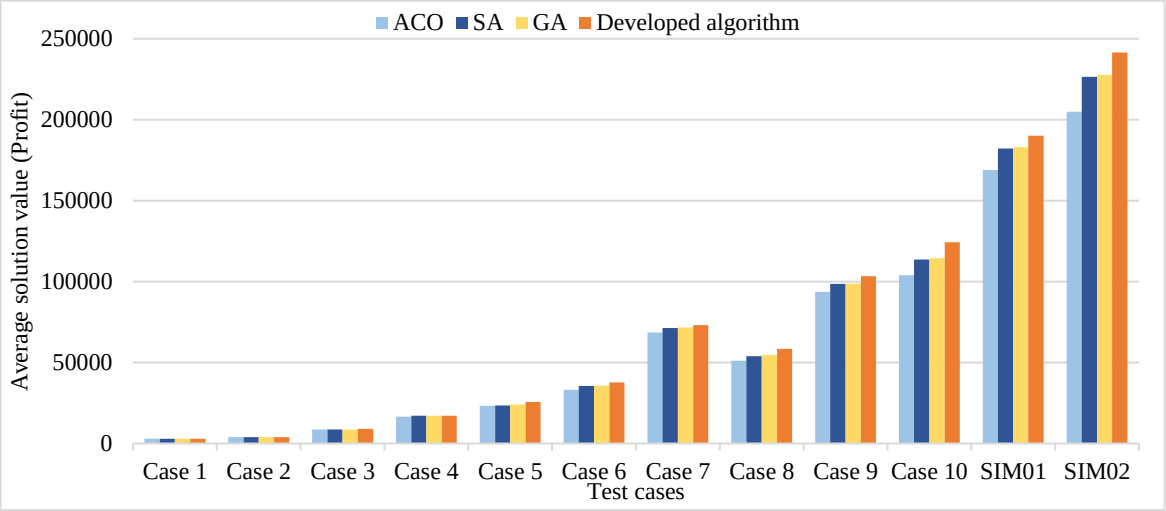

Figure 11: The average solution obtained from ACO, SA, GA, and developed algorithm.

5.5 Experiments for the two models' configurations

In this section, we report the results obtained from solving the two configurations presented in section 4 using the proposed algorithm. The statistics used in this section are Z_{best} and $\bar{Z}$ for each separate configuration in Table 7. The improvement ratio is calculated to show the effect of considering the constraints given in Eq. (10) and (11) on the profit margin. The results in figure 13, show that, in all the cases, the first configuration provides a better solution than those obtained using the second configuration. The improvement ratio starts from 2.53% in case 1, and increases up to 5.63% in case SIM02. The main reasons behind this improvement are: (1) By performing the maintenance check only if the cumulative flying hours reach the maximum level restricting the number of maintenance visits, which in turn leads to reduction of the overall maintenance cost; (2) Considering man-power availability helps the planners avoid scheduling more aircraft to maintenance stations with insufficient man-power. So, there is no reason to wait or ask for extra man-power, and this invariably reduces the penalty cost. Therefore, it is important to consider maximum flying hours and man-power availability constraints in order to reduce the total cost and maximize the profit.

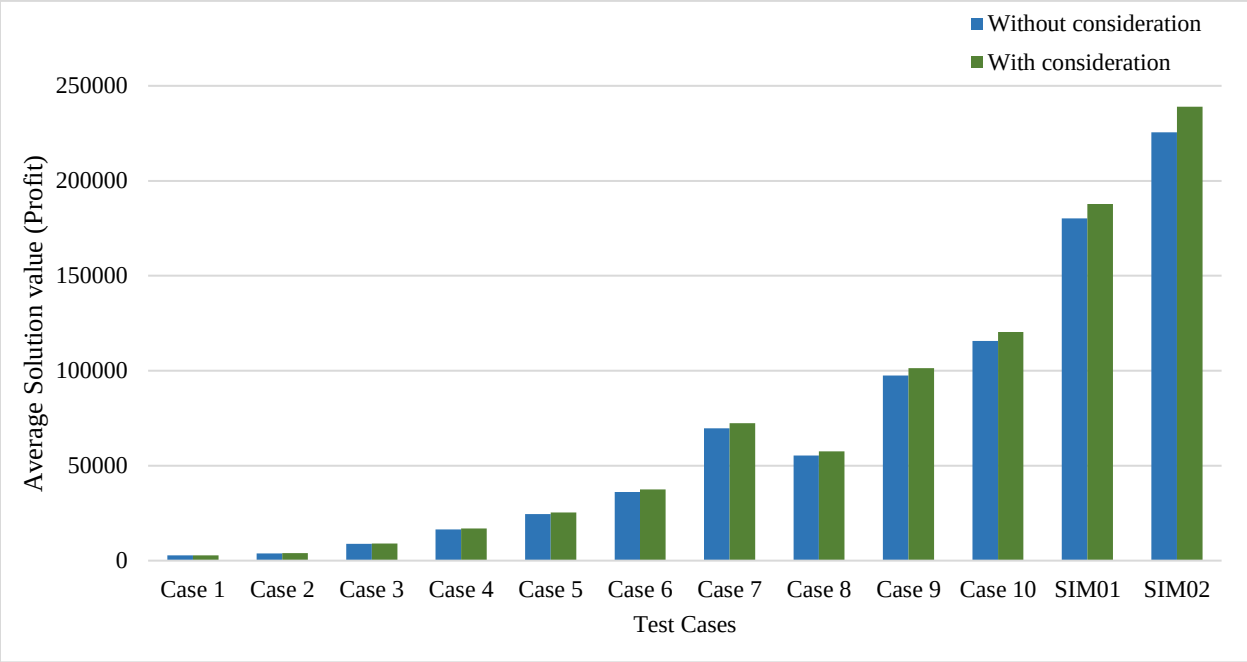

Figure 12: The Average solution obtained from first and second configurations

6. Conclusions and future directions

In this paper, we present an AMRP model that arranges feasible routes for individual aircraft, without any approximation assumption regarding the maintenance issue, and without neglecting the man-power availability consideration. To solve the proposed model, ACO, SA, and GA were proposed due to their fast response and their efficient performance in solving many NP-hard problems. In an attempt to improve the performance of the proposed heuristics, we develop a new algorithm to solve this model. The computational experiments were conducted based on real data instances obtained from the EgyptAir carrier. The computational results show that our algorithm outperforms the other three approaches in

finding a solution that does not only yields better profitability, but also results in a small gap from the upper bound. Moreover, the developed algorithm succeeded in finding the solution within 3 seconds in all the 12 real cases, which is much lesser computational time compared to ACO, SA, and GA. The model and experiments were extended to test the effect of considering the maximum flying hours and man-power availability on the profit. The results show that these considerations improve the profitability by 5.63% for the largest case. So, it is better to consider maximum flying hours than opting for one maintenance visit every four days.

With the proposed AMRP model, we have only solved the 4-day maintenance routing problem. It will be interesting to solve the weekly version of this problem, where there are demand variations between weekdays and weekends. Although our proposed model considers the man-power availability constraint, another research direction is to consider other operational constraints such as the maximum number of take-offs between two consecutive maintenance visits. In addition, integrating the proposed AMRP and other airline schedule planning phases, like FAP and CSP, would be a very fruitful research direction, since it tackles the sub-optimality issue, which means the solution is optimal in one stage and not in others. Another interesting direstion is robustness, which aims to generate routes less sensitive to disruptions. It is a pro-active way to absorb any changes that may happen in the scheduled timetable (Chung et al., 2015, Hing Kai and Alain Yee Loong, 2015). Actually, there are few research studies that pay attention to the robust AMRP, so putting much effort in this direction might be fruitfull for both academic and practitioners.

Acknowledgments

The work described in this paper was supported by grants from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. PolyU 15201414); The Natural Science Foundation of China (Grant No. 71471158); The Research Committee of Hong Kong Polytechnic University (Project Numbers G-UB03; G-YBFD; G-UA4F) and under student account code RTYN.

References

BALSEIRO, S. R., LOISEAU, I. & RAMONET, J. 2011. An Ant Colony algorithm hybridized with insertion heuristics for the Time Dependent Vehicle Routing Problem with Time Windows. *Computers & Operations Research,* 38**,** 954-966.

BAÑOS, R., ORTEGA, J., GIL, C., FERNÁNDEZ, A. & DE TORO, F. 2013. A Simulated Annealing-based parallel multi-objective approach to vehicle routing problems with time windows. *Expert Systems with Applications,* 40**,** 1696-1707.

BAŞDERE, M. & BILGE, Ü. 2014. Operational aircraft maintenance routing problem with remaining time consideration. *European Journal of Operational Research,* 235**,** 315-328.

CHENG, C.-B. & WANG, K.-P. 2009. Solving a vehicle routing problem with time windows by a decomposition technique and a genetic algorithm. *Expert Systems with Applications,* 36**,** 7758-7763.

CHUNG, S. H., YING KEI, T. & CHOI, T. M. 2015. Managing disruption risk in express logistics via proactive planning. *Industrial Management & Data Systems,* 115**,** 1481-1509.

CLARKE, L., JOHNSON, E., NEMHAUSER, G. & ZHU, Z. 1997. The aircraft rotation problem. *Annals of Operations Research,* 69**,** 33-46.

CORDEAU, J.-F., LAPORTE, G., SAVELSBERGH, M. W. P. & VIGO, D. 2007. Chapter 6 Vehicle Routing. *In:* CYNTHIA, B. & GILBERT, L. (eds.) *Handbooks in Operations Research and Management Science.* Elsevier.

DENG, G. F. & LIN, W. T. 2011. Ant colony optimization-based algorithm for airline crew scheduling problem. *Expert Systems with Applications,* 38**,** 5787-5793.
GHOSEIRI, K. & GHANNADPOUR, S. F. 2010. Multi-objective vehicle routing problem with time windows using goal programming and genetic algorithm. *Applied Soft Computing,* 10**,** 1096-1107.
GOPALAN, R. & TALLURI, K. T. 1998. The Aircraft Maintenance Routing Problem. *Operations Research,* 46**,** 260-271.
GROBA, C., SARTAL, A. & VÁZQUEZ, X. H. 2015. Solving the dynamic traveling salesman problem using a genetic algorithm with trajectory prediction: An application to fish aggregating devices. *Computers & Operations Research,* 56**,** 22-32.
HAOUARI, M., SHAO, S. & SHERALI, H. D. 2012. A Lifted Compact Formulation for the Daily Aircraft Maintenance Routing Problem. Transportation Science, 47, 508-525.
HASHIM, H. A. & ABIDO, M. A. 2015. Fuzzy Controller Design Using Evolutionary Techniques for Twin Rotor MIMO System: A Comparative Study. Computational Intelligence and Neuroscience, 2015, 11.
HASHIM, H. A., EL-FERIK, S. & ABIDO, M. A. 2015. A fuzzy logic feedback filter design tuned with PSO for adaptive controller. Expert Systems with Applications, 42, 9077-9085.
HING KAI, C. & ALAIN YEE LOONG, C. 2015. Invited review paper on managing disruption risk in express logistics via proactive planning. *Industrial Management & Data Systems,* 115.
KABBANI, N. M. & PATTY, B. W. Aircraft routing at American airlines. In Proceedings of the 32nd annual symposium of AGIFORS, 1992 Budapest, Hungary.
KIRKPATRICK, S., GELATT, C. D. & VECCHI, M. P. 1983. Optimization by Simulated Annealing. *Science,* 220**,** 671-680.
LIANG, Z. & CHAOVALITWONGSE, W. A. 2012. A Network-Based Model for the Integrated Weekly Aircraft Maintenance Routing and Fleet Assignment Problem. *Transportation Science,* 47**,** 493-507.
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**,** 109-120.
LUČIC, P. & TEODOROVIC, D. 1999. Simulated annealing for the multi-objective aircrew rostering problem. *Transportation Research Part A: Policy and Practice,* 33**,** 19-45.
MAK, V. & BOLAND, N. 2000. Heuristic approaches to the asymmetric travelling salesman problem with replenishment arcs. *International Transactions in Operational Research,* 7**,** 431-447.
OZDEMIR, H. T. & MOHAN, C. K. 2001. Flight graph based genetic algorithm for crew scheduling in airlines. *Information Sciences,* 133**,** 165-173.
POTVIN, J.-Y. & ROUSSEAU, J.-M. 1995. An Exchange Heuristic for Routeing Problems with Time Windows. *J Oper Res Soc,* 46**,** 1433-1446.
SARAC, A., BATTA, R. & RUMP, C. M. 2006. A branch-and-price approach for operational aircraft maintenance routing. *European Journal of Operational Research,* 175**,** 1850-1869.
SOUAI, N. & TEGHEM, J. 2009. Genetic algorithm based approach for the integrated airline crew-pairing and rostering problem. *European Journal of Operational Research,* 199**,** 674-683.
SRIRAM, C. & HAGHANI, A. 2003. An optimization model for aircraft maintenance scheduling and re-assignment. *Transportation Research Part A: Policy and Practice,* 37**,** 29-48.
TALLURI, K. T. 1998. The Four-Day Aircraft Maintenance Routing Problem. *Transportation Science,* 32**,** 43-53.
WANG, C., MU, D., ZHAO, F. & SUTHERLAND, J. W. 2015. A parallel simulated annealing method for the vehicle routing problem with simultaneous pickup–delivery and time windows. *Computers & Industrial Engineering,* 83**,** 111-122.
WU, Z., ZHAO, N., REN, G. & QUAN, T. 2009. Population declining ant colony optimization algorithm and its applications. *Expert Systems with Applications,* 36**,** 6276-6281.

YU, B. & YANG, Z. Z. 2011. An ant colony optimization model: The period vehicle routing problem with time windows. *Transportation Research Part E: Logistics and Transportation Review,* 47**,** 166-181.

YUAN, S., SKINNER, B., HUANG, S. & LIU, D. 2013. A new crossover approach for solving the multiple travelling salesmen problem using genetic algorithms. *European Journal of Operational Research,* 228**,** 72-82.

ZHOU, C., ZHANG, P., ZANG, W. & GUO, L. 2015. On the Upper Bounds of Spread for Greedy Algorithms in Social Network Influence Maximization. *IEEE Transactions on Knowledge and Data Engineering,* 27**,** 2770-2783.

Table 1: The configurations of our developed model versus models in the literature.

	First configuration (with consideration)	Second configuration (without consideration)
Objective function	$Max\ Z = \sum_{k\in K}\sum_{i\in NF}\left(\sum_{j\in NF} v_{ij}x_{ijk} + \sum_{j\in NFM} v_{ij}y_{ijk}\right) - \sum_{k\in K}\sum_{i\in NF}\sum_{j\in NFM} MC_k * y_{ijk} - \sum_{k\in K}\sum_{i\in NF}\sum_{j\in NFM} \alpha_k * PC_k * y_{ijk}$	
Constraints		
• Coverage (Eq. 2,3, and 4)	• √	• √
• Balance (Eq. 5,6, and 7)	• √	• √
• Time (Eq. 8)	• √	• √
• Place (Eq. 9)	• √	• √
• Max. flying hours (Eq. 10)	• √	• ***X*** and becomes one maintenance visit every 4-days.
• Man-power and maintenance (Eq. 11, 12, 13, 14, and 15)	• √	• ***X***
• Integrality (Eq. 16 and 17)	• √	• √
Solution method	The developed algorithm	

√: it means that the constraint is considered, whereas, ***X***: means that the constraint is neglected.

-

Table 2: Characteristics of all test cases.

Test cases	Number of flight legs	Fleet size	Number of airports	Maintenance Stations
Case 1	40	6	4	1
Case 2	48	7	5	1
Case 3	64	8	7	1
Case 4	96	14	13	1
Case 5	120	13	8	1
Case 6	160	11	10	1
Case 7	200	15	8	2
Case 8	240	26	19	2
Case 9	296	30	26	2
Case 10	400	42	28	3
SIM01	496	45	33	4
SIM02	696	72	53	5

Table 3: Parameters setting for ACO, SA, and GA.

ACO	• Pheromone trail importance (α)= 1
	• Heuristic function importance (β)=2
	• Number of ants=Number of nodes
	• Exploration threshold $(q_0) = 0.95$
	• Evaporation rate (ρ)=0.05
	• Control factor for pheromone laying (Q)=0.00001
	• Maximum number of iteration=1000
	• Stopping criteria
	  ○ Exceeding the max. number of iteration.
	  ○ No solution improvement over consecutive 100 iterations.

SA	• Cooling rate r=0.85 • Initial temperature T_o=100 • Maximum number of iteration=1000 • Stopping criteria ○ Same as ACO.

GA	• Randomness threshold tr=0.99
	• Population size = 60.
	• Generation number = 700.
	• Crossover rate = 0.80.
	• Mutation rate = 0.20.
	• Stopping criteria
	  ○ Exceeding the generation number.
	  ○ No solution improvement over consecutive 100 generations.

Developed Algorithm	• Stopping criteria
	  ○ Same as ACO
	  ○ Reaching the upper bound.

Table 4: The performance characteristics of ACO, SA, and GA.

Test cases	UB	**Ant colony optimization**			**Simulated annealing**			**Genetic algorithm**		
		Z_{best}	$\bar{Z}$	Gap (%)	Z_{best}	$\bar{Z}$	Gap (%)	Z_{best}	$\bar{Z}$	Gap (%)
Case 1	2857	2857	2857	0	2857	2857	0	2857	2857	0
Case 2	4000	4000	4000	0	4000	4000	0	4000	4000	0
Case 3	9143	8929	8700.72	4.83	8929	8706.44	4.77	8943	8750.27	4.29
Case 4	17143	17143	16621.71	3.04	17143	17113.29	0.17	17143	17143	0
Case 5	25714	23871	23381.43	9.07	24971	23857.44	7.22	24114	24060.29	6.43
Case 6	37714	33914	33220.86	11.91	36043	35566.43	5.69	36243	35651.13	5.46
Case 7	73143	69171	68581.71	6.23	72071	71277.71	2.55	72257	71634.86	2.06
Case 8	58429	51871	51236.81	12.30	54686	54000.63	7.57	55343	54684.71	6.40
Case 9	103429	94929	93625.6	9.47	98857	98587	4.68	98943	98843.43	4.43
Case 10	124429	104307	103983.6	16.43	115143	113635.1	8.67	115229	114362.5	8.09
SIM01	190286	171429	168901.4	11.23	184029	182300	4.19	184100	182968.9	3.84
SIM02	241571	202047	204971.4	15.15	228029	226460	6.25	228100	227741.9	5.72

Table 5: Computational time and iteration for stopping for ACO, SA, and GA.

Test cases	**Ant colony optimization**		**Simulated annealing**		**Genetic algorithm**	
	$CPU(s)$	Iterations for stopping	$CPU(s)$	Iterations for stopping	$CPU(s)$	Iterations for stopping
Case 1	1.30	100	0.87	100	12.28	100
Case 2	1.73	105	1.23	100	15.05	100
Case 3	4.85	205	2.07	152	31.40	160
Case 4	6.50	177	3.62	136	44.40	100
Case 5	9.93	178	9.08	265	56.01	123
Case 6	24.29	295	11.82	266	81.99	193
Case 7	29.29	246	16.32	250	204.51	184
Case 8	38.07	244	33.91	413	326.45	194
Case 9	63.87	300	43.59	354	368.18	156
Case 10	147.98	399	81.95	444	1428.51	324
SIM01	244.12	478	121.40	418	1203.14	201
SIM02	380.02	361	333.63	608	2371.59	194

Table 6: Performance characteristics of the developed algorithm.

Test cases	**The developed algorithm**							
	Z_{best}=UB	$\bar{Z}$	Gap (%)	$CPU(s)$	Iterations for stopping	IR over ACO	IR over SA	IR over GA
Case 1	2857	2857	0	0.19	1	0	0	0
Case 2	4000	4000	0	0.20	1	0	0	0
Case 3	9143	**9143**	0	0.21	1	4.83	4.77	4.29
Case 4	17143	17143	0	0.23	1	3.04	0.17	0
Case 5	25714	**25714**	0	0.26	1	9.07	8.81	6.43
Case 6	37714	**37714**	0	0.29	1	11.91	5.69	5.46
Case 7	73143	**73088**	0.075	0.37	4	6.16	2.47	1.98
Case 8	58429	**58400**	0.049	0.47	2	12.26	7.53	6.36
Case 9	103429	**103407.1**	0.021	0.62	2	9.45	4.66	4.41
Case 10	124429	**124385.71**	0.034	0.79	2	16.40	8.64	8.05
SIM01	190286	**190204.3**	0.042	1.21	3	11.20	4.15	3.80
SIM02	241571	**241523.6**	0.019	2.84	3	15.13	6.23	5.70
Average improvement ratio =						8.30%	4.45%	4.00%

IR: Improvement ratio, $IR = \left(\bar{Z}_{developed\ algorithm} - \bar{Z}_{heuristic}\right)/\bar{Z}_{developed\ algorithm}$, The heuristic is ACO, SA, and GA.

Table 7: The two configurations results obtained by using the developed algorithm

Test cases	**First configuration (With consideration)** **Our proposed model**		**Second configuration (Without consideration)** **Literature models**		Imp. ratio (%)
	Z_{best}	$\bar{Z}$	Z_{best}	$\bar{Z}$	
Case 1	2821	2821.42	2750	2750	2.53
Case 2	3964	3964.28	3857	3857.14	2.70
Case 3	9107	9076.48	8896	8814.82	2.88
Case 4	17061	16970.55	16539	16474.07	2.92
Case 5	25714	25293.89	24914	24475.93	3.23
Case 6	37700	37498.5	36750	36145.93	3.60
Case 7	72993	72313.57	70279	69651.79	3.68
Case 8	58157	57525.57	55729	55354.36	3.77
Case 9	102686	101394.3	97936	97407.14	3.93
Case 10	122186	120417.7	116214	115640.1	3.97
SIM01	189236	187828.6	181286	180276.7	4.02
SIM02	240471	238932.6	226300	225476.3	5.63

Imp. ratio (%) $= \bar{Z}_{with} - \bar{Z}_{without} / \bar{Z}_{with}$