A Model for Robust Aircraft Maintenance Routing Problem

Ru-yu CUI*, Xing-ye DONG and You-fang LIN

Beijing Key Lab of Traffic Data Analysis and Mining, School of Computer and Information Technology, Beijing Jiaotong University, Beijing 100044, China

*Corresponding author

Keywords: Aircraft Maintenance Routing, Robustness, VNS.

Abstract. An MIP model is established for the aircraft maintenance routing problem. In this model, two objectives, i.e., minimizing the number of used aircrafts and total flight delay costs, are considered. In terms of solution method, a VNS is proposed and experimental results show that it can find much better solutions quickly in reasonable time than CPLEX.

Introduction

The airline operational scheduling determines whether the operations can be carried out smoothly, as well as the costs and benefits and the market competitiveness of airlines. Therefore, optimal operational scheduling is of great significance. In this paper, we focus on aircraft maintenance routing problem (AMRP), which is a sub problem of airline operational scheduling problem. Many scholars [1,2] have done a lot of researches on it. Talluri et al. [3] propose a polynomial algorithm to solve the 4-day AMRP, and first prove this problem is NP-hard. Başdere et al. [4] formulate an integer linear programming model. Al-Thani et al. [5] modify the model proposed by Başdere et al. [4] and simplify the complexity of the model. However, it lacks the discussion of robustness in literature. Robust airline flight scheduling refers to aircraft scheduling plan itself has certain capacity of resisting disturbance, i.e., adjusting the cushion times in the aircraft scheduling to reduce flight delays and cancellations. If the scheduling is weak of robustness, the delay spread of a flight likely leads to a large area of flight delays. In this work, we put forward a robustness AMRP model based on delay probability, with the consideration of the flight delay costs.

The rest of the paper is organized as follows. In Section 2, we propose the mixed integer mathematical model. Section 3 introduces a heuristic algorithm to find good solution quickly for large-scale problems. In section 4, the computational results carried out on real data are summarized. Finally, the paper is concluded in Section 5.

The Proposed Model

In this section, the problem is firstly described as a connection network derived the representation by Ahmed et al [2], then illustrate the definiton of the delay probability, finally follows the proposed MIP.

Connection Network

Let $s$ be the source node, $t$ be the sink node, $K$ be the set of nodes representing aircrafts, $L$ be the set of flight leg nodes, and let $V = \{s\} \cup K \cup L \cup \{t\}$. Arc set $A$ includes four mutually exclusive sub sets: $A^S, A^K, A^L, A^T$. The sub set $A^S$ includes $|K|$ arcs $(s,k)(k \in K)$, where each arc outgoes from $s$ to an aircraft node $k$. Set $A^K$ consists of arcs $(k,j)(k \in K, j \in L)$ satisfying the constraint that the initial location of the aircraft $k$ should be the same as the departure airport of the flight $j$ and the departure time of the flight $j$ should less than 24 h from the start of the planning horizon. Arcs $(i,j)(i \in L, j \in L)$ belongs to $A^L$, if flight leg $i$ and $j$ are compatible, i.e. the arrival airport of flight leg $i$ is the same as the departure airport of the flight leg $j$ and the connection time between them is not less than the minimum aircraft turnaround time and less than 24 h. The sub set $A^T$ includes arcs $(j,t)(j \in L)$ from the node $j$ to node $t$ where the arrival time of $j$ is less than 24 h to the end of the planning horizon.
The Definition of the Delay Probability

Based on the analysis of historical data and experience, the probability of flight delay can be inferred by the length of connection time between two consecutive flights [6], just shown as Fig. 1.

\[
\begin{array}{c|c|c|c|c}
0 & t_1 & m_1 & t_2 & m_2 & \cdots & t_k & m_k & t_{k+1} & \text{min}
\end{array}
\]

Figure 1. The relationship between connection time and flight delay probability.

In Fig. 1, \(t_i (i = 1, 2, \ldots, k + 1)\) is the connection time between two consecutive flights, \(t_1\) is the minimum turnaround time, and \(m_i (i = 1, 2, \ldots, k + 1)\) is the probability of flight delay when flight connection time is between \(t_i\) and \(t_{i+1}\). When the flight connection time is more than \(t_{k+1}\), the delay probability is \(m_{k+1}\). Then the delay probability of flight \(i\) in route can be defined as Eq. (1).

\[
\text{pr}_i = \begin{cases} 
0, & n = 0 \\
1 - \prod_{r=1}^{n}(1 - q_r), & n \geq 1
\end{cases}
\]

(1)

where \(n\) is the number of flights before the flight \(i\) in aircraft route and \(q_r = m_i\) refers to the delay probability of the \(r\)th connection.

Let \(a_{ij}\) be the delay probability based on the connection time between flight \(i\) and flight \(j\), then the delay probability \(pr_j\) can be computed as \(a_{ij} + (1 - a_{ij})pr_i\) according to the Theorem 1.

**Theorem 1.** Suppose that flight \(i\) is the predecessor flight of flight \(j\), then \(pr_j = a_{ij} + (1 - a_{ij})pr_i\).

**Proof.** According to the Eq.(1) and the hypothetical condition, it can be drawn that \(pr_i = 1 - \prod_{r=1}^{n}(1 - q_r)\) and \(pr_j = 1 - \prod_{r=1}^{n+1}(1 - q_r)\). On the basis of the definition of \(a_{ij}\), we can get \(q_{n+1} = a_{ij}\), so \(pr_j = 1 - \prod_{r=1}^{n+1}(1 - q_r) = 1 - (1 - a_{ij}) \prod_{r=1}^{n}(1 - q_r) = 1 - \prod_{r=1}^{n}(1 - q_r) + a_{ij} \prod_{r=1}^{n}(1 - q_r) = pr_i + (1 - pr_i)a_{ij} = pr_i + a_{ij} - pr_i a_{ij} = a_{ij} + (1 - a_{ij}) pr_i\).

MIP Model

Firstly, the following notions are introduced. The proposed MIP is derived from the work of Al-Thani et al. [5], but with the extension of minimizing the number of aircrafts and the robustness of model.

**Set Definition:**

- \(\delta_i^- / \delta_i^+\): set of arcs that are incoming to/outgoing from node \(i, i \in L\).
- \(A_M \subset A\): set of maintenance arcs. Compatible arc \((i, j)\) belongs to \(A_M\) if the arrival airport of flight \(i\) has the maintenance capability and the connection time between flight \(i\) and flight \(j\) is greater than or equals to the maintenance check duration.
- \(A_M^- = A \setminus A_M\): set of non-maintenance arcs.
- \(L_M \subset L\): set of maintenance nodes. Node \(j \in L_M\) if and only if it is feasible to schedule a maintenance before the departure of \(j\).

**Parameter Definition:**

- \(\alpha\): the weight of the number of the aircrafts.
- \(m\): the number of the aircrafts, i.e. the size of the aircraft set \(K\).
- \(T_i\): the delay time of flight \(i\).
- \(Cost\): delay cost per unit time.
- \(F\): the max flying time of the aircraft before a maintenance check.

**Decision variables:**

- \(x_{ij} = \begin{cases} 
1, & \text{if arc } (i, j) \in A \text{ is selected} \\
0, & \text{otherwise}
\end{cases}\)
- \(y_j = \begin{cases} 
1, & \text{if a routine check is completed before executing flight } j, j \in L_M \\
0, & \text{otherwise}
\end{cases}\)
- \(u_j\): a continuous variable representing the cumulative flying time after the execution of node \(j \in V\) since the last maintenance check.
The mixed integer programing model of the aircraft maintenance routing problem with the consideration of minimizing the number of used aircrafts and minimizing the total flight delay costs is as follows:

\[
\min \alpha \sum_{k \in K} x_{sk} + \sum_{i \in L} p_{r_i} \cdot T_i \cdot \text{Cost} \tag{2}
\]

s.t:

\[
\sum_{k \in K} x_{sk} \leq m \tag{3}
\]

\[
x_{sk} = \sum_{(k,j) \in \delta^+_k} x_{k_j}, \forall k \in K \tag{4}
\]

\[
\sum_{(i,j) \in \delta^+_i} x_{ij} = 1, \forall i \in L \tag{5}
\]

\[
\sum_{(j,i) \in \delta^-_i} x_{ji} = \sum_{(i,j) \in \delta^+_i} x_{ij}, \forall i \in L \tag{6}
\]

\[
y_j \leq \sum_{(i,j) \in A_M} x_{ij}, \forall j \in L_M \tag{7}
\]

\[
u_j \leq u_i + d_j + (F - d_i - d_j)(1 - x_{ij}), \forall (i,j) \in A \setminus \delta^-_i \tag{8}
\]

\[
u_j \leq F - (F - d_j)y_j, \forall j \in L_M \tag{9}
\]

\[
u_j \geq u_i + d_j - F(1 - x_{ij}), \forall (i,j) \in A_M \setminus \delta^-_i \tag{10}
\]

\[
u_j \geq u_i + d_j - F(1 - x_{ij}) - Fy_j, \forall (i,j) \in A_M \tag{11}
\]

\[
p_{r_j} \geq a_{ij} + (1 - a_{ij})p_{r_i} - 2(1 - x_{ij}), \forall (i,j) \in A_L \tag{12}
\]

\[
p_{r_j} \leq a_{ij} + (1 - a_{ij})p_{r_i} + 2(1 - x_{ij}), \forall (i,j) \in A_L \tag{13}
\]

\[
p_{r_j} \geq x_{kj} - 1, \forall k \in K, (k,j) \in \delta^+_k \tag{14}
\]

\[
p_{r_j} \leq 1 - x_{kj}, \forall k \in K, (k,j) \in \delta^+_k \tag{15}
\]

\[
0 \leq p_{r_j} \leq 1 \tag{16}
\]

\[
u_s = 0, u_k = f_k, \forall k \in K \tag{17}
\]

\[
x_{ij} \in \{0,1\}, \forall (i,j) \in A \tag{18}
\]

\[
y_j \in \{0,1\}, \forall j \in L_M \tag{19}
\]

\[
d_j \leq u_j \leq F, \forall j \in V \tag{20}
\]

The objective (2) is to minimize the number of used aircrafts and the total flight delay costs. Constraint (3) assures that the number of aircrafts to be used in the model should be less than or equal to the maximum number of aircrafts available. Constraint (4) underlines that every aircraft \(k\) used in the model should be followed by a successor flight node \(j\) as the first flight mission. Constraint (5) is used to guarantee that any flight node has only one successor node. Constraint (6) and Constraint (5) can derive that any flight node has only one predecessor node. Constraint (7) ensures that if the maintenance arc incoming to maintenance node \(j\) did not be chosen in the aircraft route, the maintenance check is not allowed. Constraint (8-11) realize the definition of the cumulative flying time after aircraft executing node \(j, j \in V\) since the last maintenance check. Constraint (12) and
Constraint (13) establishes the relationship between \( p_r^j \) and \( p_r^i \), if \( x_{ij} = 1 \), then \( p_r^j = a_{ij} + (1 - a_{ij})p_r^i \), otherwise the Constraint (12) and Constraint (13) become redundant. The Constraint (14) and Constraint (15) set the delay probability of first flight \( j \) carried by aircraft \( k \) to 0. Constraint (16) defines the value of \( p_r^j \) between 0 and 1. Constraint (17) initializes the cumulative flying time of the source node and aircraft node. The \( f_k \) refers to the initial cumulative flying time of aircraft \( k \). Constraint (18) and (19) define the range of decision variable \( x \) and decision variable \( y \), respectively. Constraint (20) defines inequality constraint for the cumulative flying time \( u \).

### A Heuristic Algorithm of Neighborhood Search

Variable neighborhood search is a heuristic algorithm firstly proposed by Mladenovic and Hanse [7], which has been made great progress in solving combinatorial optimization problems and global optimization problems. The essential idea of the algorithm is to expand the search scope by changing the neighborhood structure of current solution to obtain the local optimal solution for all the used neighborhood structures. The main idea of VNS for our model derived from [5] is as follows:

First, a commercial solver is used to produce a feasible solution quickly. Then a portion of the aircrafts and the corresponding flights covered by these aircrafts from the current solution are selected to form a small scale model and then rescheduled. If the objective of this small model is better than the original one, the original is replaced by the new scheduling result. The pseudo code is provided in Algorithm 1, where \( S \) is the solution of the model; \( s \) is the number of the routes selected from the \( S \), including \( s \) aircrafts, as well as a series of flights covered by these \( s \) aircrafts; \( s_{min} \) and \( s_{max} \) are the minimum and maximum allowed value of \( s \), respectively; \( q_{max} \) is the maximum iterations; \( r \) is the counter of the generations. A route can be defined that it consists of one aircraft and some flights covered by this aircraft, then the solution \( S \) can be divided into two parts, \( \epsilon \) and \( N\epsilon \), where set \( \epsilon \) includes those routes with larger objective than the median value, while \( N\epsilon \) includes the remaining.

### Algorithm 1. The improved variable neighborhood search.

**Input:** \( s_{min}, s_{max}, q_{max} \)

**Output:** \( S_{best} \)

1: function VNS(\( s_{min}, s_{max}, q_{max} \))
2:     Generate an initial solution \( S^0 \), set \( s = s_{min}, r = 1, q = 0, S_{best} = S^0 \);
3:     while \( s < s_{max} \) do
4:         while \( q < q_{max} \) do
5:             Build subset \( \epsilon \) and \( N\epsilon \);
6:             Select one route from \( \epsilon, s - 1 \) routes from \( N\epsilon \) to establish a new instance \( I^r \);
7:             Solve model \( I^r \) using a solver;
8:             while solution of \( I^r \) is improved do
9:                 Replace \( s \) old routes in \( S_{best} \) with new \( s \) routes;
10:                Set \( r = r + 1, s = s_{min}, q = 0 \);
11:               Build subset \( \epsilon \) and \( N\epsilon \);
12:              Select one route from \( \epsilon, s - 1 \) routes from \( N\epsilon \) to establish a new instance \( I^r \);
13:             Solve model \( I^r \) using a solver;
14:         end while
15:     Set \( q = q + 1 \);
16: end while
17: Set \( s = s + 1, q = 0 \);
18: end while
19: end function

### Computational Experiments

All experiments are executed on an Intel Core i5-4210U processor 1.70Ghz desktop computer running Windows 7 x64. The algorithms are coded in C++, and the commercial solver is CPLEX 12.6 (Windows x86_32). In the following experiments, we set the maximum running time to 3 h.
The data set is provided by Basdere and Bilge [4]. This data set includes 667 flights with 20 aircrafts and the difference for each instance lies in the difference of the initial airports and the initial remaining flying times.

For the parameters in the model, the aircraft minimum turnaround time is 35 min, the maximum allowed flying time is 100 h, and the maintenance time is 8 h. The average delay time for each flight is 12 minutes, and the unit delay time cost is 60 per hour. The delay probability between two flights is determined according to the connection time between the consecutive flights: if the time is greater than 35 minutes and less than or equals to 60 minutes, the delay probability is set as 80%; if the time is greater than 60 minute and less than or equals to 120 minutes, the delay probability is set as 30%; if the time is greater than 120 minutes, then the delay probability is set as 5%. With regard to the parameters in algorithm VNS, it is set that $s_{\text{min}} = 3$, $s_{\text{max}} = 5$, $q_{\text{max}} = 10$.

Table 1 displays the experimental results. The column Num refers to the number of used aircrafts, the column DelayCost refers to total delay costs and column Time is the running time of the corresponding algorithm in seconds. The average relative percentage deviation (RPD) is calculated for each instance as $\text{RPD} = \frac{\text{Time(CPLEX)} - \text{Time(VNS)}}{\text{Time(CPLEX)}}$.

<table>
<thead>
<tr>
<th>Instance</th>
<th>CPLEX</th>
<th>VNS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>OBJ</td>
<td>OBJ</td>
</tr>
<tr>
<td>Num</td>
<td>DelayCost</td>
<td>Time</td>
</tr>
<tr>
<td>1</td>
<td>20</td>
<td>7019</td>
</tr>
<tr>
<td>2</td>
<td>20</td>
<td>7071</td>
</tr>
<tr>
<td>3</td>
<td>20</td>
<td>7199</td>
</tr>
<tr>
<td>4</td>
<td>20</td>
<td>6990</td>
</tr>
<tr>
<td>5</td>
<td>20</td>
<td>7298</td>
</tr>
<tr>
<td>6</td>
<td>20</td>
<td>7082</td>
</tr>
<tr>
<td>7</td>
<td>20</td>
<td>7049</td>
</tr>
<tr>
<td>8</td>
<td>20</td>
<td>7056</td>
</tr>
<tr>
<td>9</td>
<td>20</td>
<td>7173</td>
</tr>
<tr>
<td>10</td>
<td>20</td>
<td>7006</td>
</tr>
<tr>
<td>11</td>
<td>20</td>
<td>7066</td>
</tr>
<tr>
<td>12</td>
<td>20</td>
<td>7053</td>
</tr>
<tr>
<td>13</td>
<td>20</td>
<td>7054</td>
</tr>
<tr>
<td>14</td>
<td>20</td>
<td>7063</td>
</tr>
<tr>
<td>15</td>
<td>20</td>
<td>7107</td>
</tr>
<tr>
<td>16</td>
<td>20</td>
<td>7198</td>
</tr>
<tr>
<td>17</td>
<td>20</td>
<td>7107</td>
</tr>
<tr>
<td>18</td>
<td>20</td>
<td>7099</td>
</tr>
<tr>
<td>19</td>
<td>20</td>
<td>7141</td>
</tr>
<tr>
<td>20</td>
<td>20</td>
<td>7103</td>
</tr>
</tbody>
</table>

It can be seen from Table 1 that CPLEX failed to obtain a better solution with the limits on maximum computing time in terms of total delay costs while a better solution is obtained by the VNS algorithm with fewer time. So, the VNS algorithm is superior to CPLEX in terms of computation time and objective function, especially in computing time.

**Summary**

In this paper, an MIP model is established aiming at generating a robust aircraft maintenance routing problem and by using less aircrafts. A VNS is designed. Experimental results show that the proposed model is feasible and the VNS can find better solutions than CPLEX in much shorter time.
References


