A Weight-Based Graph Coloring Approach to Airport Gate Assignment Problem

Yanjun Jiang and Xueyan Song

School of Computer Science & Technology, Tianjin University, Tianjin 300072

Abstract. A graph coloring model of airport gate assignment problem (AGAP) is constructed in this paper, and a kind of new time slot algorithm is used to find out the conflict sets of all the flights. By considering both the type and time conflict of a flight comprehensively, a new criterion for measuring the difficulty of a flight is put forward which can be used to compute the difficulty coefficients of each flight to be assigned. And after that all these flights will be sorted in descending order by their difficulty coefficients before assigning them to available gates. Finally, experimental results on the stochastic data sets demonstrate the effectiveness of the algorithm put forward in this paper.

Keywords: Gate Assignment, Vertex Coloring, Difficulty Coefficients.

1 Introduction

With the rapid development of society and economy, there is a growing demand for air travelling, which greatly enlarges the scale of civil aviation and results in an increasing number of flights. However, the construction of airport gates is far lagged behind. Therefore, the contradiction between the growing number of flights and the inadequacy of gates becomes more and more severe. Under the circumstances, we need a more excellent algorithm to address the optimization problem of assigning fast increasing number of flights to limited number of airport gates. And numerous scholars have done great amount of research work in this field. Oversea Researchers have a profound understanding of this problem and have obtained a multitude of outstanding optimization results. To sum up, their researches come down to three categories: Mathematical Programming Approach [1-2], Expert System [3-4] and System Simulation [5-6]. In China, scholars began doing research in this filed much later but we have also achieved some excellent research results. For instance, Xueming Zhang and Fazhong Shi [7] put forward and implemented Three-Level-Reasoning -Model Airport Gate Automatic Expert System; Chen Tian, Guixi Nai [8] utilized the Genetic Algorithm to solve the AGAP; Jun Wen, Bing Li[9] etc. introduced the time slot division approach to define the clash sets of flights and they also converted the AGAP to graph coloring problem; Rongwu Luo, Ruhe Xie[10] did further research on the results of Wen & Xie, and they put forward an improved time slot algorithm and use a new division method and coloring algorithm to optimize the assignment results. In this paper, we do further research on the basis of Wen and Luo's research results and we propose an innovate time slot division approach and bring in the new concept of Difficulty Coefficients, which produce better results of AGAP than theirs.

2 The Graph Coloring Model for AGAP

2.1 AGAP

According to the size of flights, we can divide the flights into three categories: large size, medium size and small size. Similarly, we can separate the gates in the airport to three categories: the large type, the middle type and the small type. The AGAP means that assigning appropriate gates for all the arriving and leaving flights at an airport to ensure all the flights run on time. This issue belongs to constraint optimization problems and it is NP-Complete, thus no simple polynomial time algorithm for it.

2.2 Constraints for AGAP

The constraints that need to be satisfied when assigning a gate are as follows:

1. There can be at most one flight on one gate at the same time interval.
2. Every flight must be assigned to one gate, and also at most one gate.
3. The minimum grounding time constraint must be satisfied. That is, the time interval between the flight arriving and leaving should be longer than the minimum ground time.
4. The successive flights at the same gate should keep a safe time interval to secure flights entering and leaving smoothly (this is a soft constraint).
5. The flights and gates mutual matching constraint should be satisfied. In other words, the large flights can only be assigned to large gates, middle flights can be assigned to both large and middle gates, and small flights can be assigned to any type of gates.

2.3 Graph Coloring Model for AGAP

The AGAP can be converted into an undirected graph coloring optimization problem [11]. In this problem, the flights are viewed as vertexes in the graph and gates are considered as the available colors. If there is time conflict between two flights then there will be an edge between the two vertexes in the graph. The optimization problem in this paper is to use the fewest gates to park all the flights. In other words, our aim is to find the coloring number of the undirected graph. This graph coloring model can be illustrated in the following formal language. $G(V, E)$ where, $V = \{ v_i$ | all the flights need to be assigned $i \}$, $E = \{(v_i, v_j)$ | if there is time conflict between vertexes $v_i, v_j \}$

3 Algorithm for AGAP

The objective function in this paper is to minimize the number of used gates under the condition of that all the flights can be assigned a gate. If the number of gates used is the same, we would consider the one use smaller gates to be better. Since the number of gates in an airport is certain, thus this minimum problem can be transformed into a maximum problem — maximizing the remaining gates number. We make the following assumption. The number of remaining large, medium and small type gates are denoted as x、 y、 z. As it is better to leave behind large type of gates than medium and small type gates, therefore, in order to measure the assignment results more precisely, we give different priority to different type of gates. There are many different priority assignment methods, in this paper, we use the simplest approach. We set the priority of large, medium and small type gates to 3, 2 and 1 separately. The linearity-weighted-sum of remaining gates is utilized as the objective function which is formulated as follows:

$$\text{maximize } \operatorname{Score}(x,y,z) = x + 2y + 3z \tag{1}$$

The AGAP assignment algorithm is divided into three steps:

1. Define the time conflicts set between all the flights.
6. Sort all the flights according to their difficulty coefficients.
7. Color the vertexes in the undirected graph.

3.1 Algorithm for Defining the Time Conflicts Set between All the Flights

Inspired by the method put forwarded in the paper [10], we designed an innovative approach for defining the conflict time set. The new method does not need to sort all the flights according to their arriving and departing time. So it is easy to be implemented. Assuming that all the flights set need to be assigned is $V = \{v_1, v_2, \dots v_N\}$, here N is an integer which signifies the number of flights to be assigned. b_i is used to indicate the beginning time of flight i, and all the b_i compose the beginning time set $B = \{b_i \mid i = 1, 2, .. N\}$. Similarly, the finishing time of flights i is denoted by f_i, and the finishing time set $F = \{f_i \mid i = 1, 2, .. N\}$ is comprised of all the f_i. All the flights that conflict with flight i ($j > i$) is signified as $S_i (i = 1, 2, .. N)$, $S = S_1 \cup S_2 \cup \ldots \cup S_N$.

```
Algorithm 1. Pseudo-code
Input: B, E
for $= i$ to $i <= N$
   for $= j$ to $j <= N$
      if $b_i == b_j$
         add $v_j$ $to$ $S_i$
      else if $b_i < b_j$
         if $f_i > b_j$
            add $v_j$ $to$ $S_i$
      else if $b_i > b_j$
         if $b_i < f_j$
            add $v_j$ $to$ $S_i$
   end for
end for
Output: $S$
```

Fig. 1. Algorithm for defining the time conflicts set

The defining method for conflicts between flight time is as follows: if two flight i and flight j arrive at the same time then they conflict with each other thus they cannot be scheduled to the same gate at the same time; if the arriving time of flight i is between the arriving time and departing time of flight j then they also conflict; if the arriving time of flight j is between the arriving time and departing time of flight i which also means that there is conflict between them.

The pseudo code of this algorithm is listed in Fig.1

3.2 Sorting Algorithm of Flights Difficulty Coefficients

In this paper, we would assign the flights according to their difficulty coefficients, which depend on two aspects: the number of conflicts of the flight with others and the size of this flight. Namely, large size flights are more difficult to assign than medium size flights and medium size flights are more difficult to assign than small size flights. In paper [9]' the author assign flights according to their conflicts number, if two flights have the same conflicts number then the size of flights will be considered. This means that conflicts number is more important than flights size. Conversely, paper [10] gives more priority to size of flights, they first schedule flights according to their size, if two flights are the same size then they will consider their conflicts number. However, according to our experiments and analysis, we believe that both two methods have advantages and disadvantages. They adapt to different flights data set. By considering both sides comprehensively, we put forward an innovative method for measuring difficulty of assigning flights. The difficult coefficients are formulated as follows:

$$difficulty_i = \alpha \frac{type_i}{max\_type} + (1-\alpha) \frac{degree_i}{max\_degree} \tag{2}$$

where:

$difficulty_i$: The difficulty of flight i;

$\alpha (0 \le \alpha \le 1)$: the weight allocated to the size of flights;

$type_i$: The type flight i, and:

$$type_i = \begin{cases} 1 & \text{flight } i \text{ is small type} \\ 2 & \text{flight } i \text{ is medium type} \\ 3 & \text{flight } i \text{ is large type} \end{cases}$$

max_type: Value of largest flight;

$degree_i$: Conflicts number of flight i with other flights;

max_degree: The maximum of time conflicts number for all flights.

We can assign different weight to the conflicts number and type of flights by adjusting the value. Therefore, we can change the priority for assigning all the flights.

From the formulation of definition (2), we know that the less the value, the less important of flights type but more important of conflicts number, which is suitable for flights with more conflicts number and smaller size but adverse to flights with less conflicts and larger size, and vice versa. By changing value dynamically, we can find a much better results for different flights set. The pseudo code for sorting algorithm of flights difficulty coefficients is listed below.

```
Algorithm 2. Pseudo - code
Calculate the degree of each flight
for $k$ = 1 to $k <= N$
    for $i$ = 1 to $i <= N$
    if $degree_i$ > max_degree
        max_degree = $degree_i$
    end for
    for $i$ = 1 to $i <= N$
        $difficulty_i = \alpha * type_i +$
            $(1-\alpha) * \text{degree}_i / \text{max\_deg}ree$
    end for
    sort all the remaining vertexes according
        to their difficulty coefficients
    assign the sequence number $k$ to the flight
        with maximum difficulty coefficients
    subtract one to the degrees of all its adjacent vertexes
    recalculate the degree of each flight
end for
```

Fig. 2. Graph 2, sorting algorithm of flights difficulty coefficients

3.3 Graph Coloring Algorithm of AGAP

We can calculate all the sequence number using the method put forward in 3.2, and then we will sort all the flights according to their sequence number and then we obtain the assigning order for all flights, from 3.2 we know that the order is the descending order of flights assignment difficulty coefficients. Thereafter, the algorithm proposed in paper [10] is utilized to color all the flights vertexes. So as to search the best assignment, different α value is used to adjust the weight between the two aspects of flights assignment. Therefore, the best result can be chosen during the variation of α .

4 Experimental Results

4.1 Experimental Data Set Generation

In order to validate the new algorithm that we put forward, ten different data sets are collected to compare the results among paper [9], [10] and ours. The flights data sets are randomly generated by computer program. And each data set is composed of flights data and gates data.

1. **Flights data sets generation method.** By setting 5 minutes as the time unit for flights, the arriving times of flights are produced firstly, which follow uniform distribution between interval [0, 48]. This can be depicted by formal language, that is, assuming X is an integral random variable, obeying the uniform distribution between interval [0, 48], a value for X is generated randomly at first and then 5*X is used as the arriving time for the flight. Similarly, the parking time for flights can be obtained. In our experiment, we assume that the flights parking time is between 30 minutes and 90 minutes. So if Y is denoted the integral random variable for parking time which follow the uniform distribution between interval [6, 18], a value for Y is generated randomly at first then 5*Y is used as the parking time for the flight. Finally, the departing time of the flight can be acquired by adding the parking time to beginning time.
8. **Gate data generation method.** Each flight has two attributes: the gate number and the gate type. The former can be obtained by giving number to the flights according to their generation sequence. The gate type can be gotten by randomly choosing from 1, 2 and 3.

4.2 Experimental Results

The symbol "--" in the above table signifies no solution can be found. The best results in the table are in bold typeface. From the table, we can conclude that algorithms in paper [9] and [10] have its own advantage, but our method is better than both of theirs. Therefore, we know from the experimental results that the best choice for the AGAP is not to give too much priority to each aspects of the problem, but to consider them comprehensively and select the best results through adjusting the weight factor.

Table 1. Experimental Results

Data set	Flight num	Gate num	[9]	[10]	ours
Data 1	50	25	5	--	6
Data 2	60	30	21	20	23
Data 3	80	40	29	38	39
Data 4	100	50	24	31	31
Data 5	100	50	--	--	34
Data 6	100	60	46	55	58
Data 7	100	70	77	78	80
Data 8	120	60	19	28	28
Data 9	200	100	68	79	84
Data 10	200	100	81	81	85

The results are based on the data sets randomly generated using the method in 4.1.

The symbol "--" in the above table signifies no solution can be found. The best results in the table are in bold typeface. From the table, we can conclude that algorithms in paper [9] and [10] have its own advantage, but our method is better than both of theirs. Therefore, we know from the experimental results that the best choice for the AGAP is not to give too much priority to each aspects of the problem, but to consider them comprehensively and select the best results through adjusting the weight factor.

5 Conclusion

In this paper, we conduct further research on the forefathers in this field and establish a graph coloring method for AGAP. What's more, a new comprehensive method of measuring flights assignment difficulty is initially used. By adjusting the α value, we can find a most suitable value for each data set so as to minimize the gates number occupied. And the experimental results confirm that our algorithm is superior than the method used in [9],[10], which means our method can boost the flights utilization rate and reduce the operating fees of airport.

Acknowledgement. This study is supported by grants from National Natural Science Foundation of China and Civil Aviation Administration of China(61039001) and Tianjin Municipal Science and Technology Commission(11ZCKFGX04200).

References

1. Xu, J.F., Bailey, G.: The Airport Gate Assignment Problem: Mathematical Model and a Tabu Search Algorithm. In: Proceedings of the 37th Hawaii International Conference on System Sciences (2001)
2. Ding, H., Lim, A., Rodrigues, B.: Aircraft and Gate Scheduling Optimization at Airports. In: Proceedings of the 37th Hawaii International Conference on System Sciences (2004)
3. Geun-Sik, J., Jong, J.J., Chang, Y.Y.: Expert System for Scheduling an Airline Gate Allocation. Expert Systems with Application 13(4), 275–282 (1997)
4. Soi, H.L., Jia, M.C., Henry, F.: Development of an Intelligent Agent for Airport Gate Assignment. Journal of Air Transportation 7(2), 103–114 (2002)
5. Cheng, Y.: A Ruled-Based Reactive Model for the simulation of Aircraft on Airport Gate. Knowledge-Based Systems 10(4), 225–236 (1998)
6. Chang, Y.: A Network Model for Gate Assignment. Journal of Advanced Transportation 32(2), 176–189 (1998)
7. Xue, M.Z., Shi, F.Z.: Research on Airport Gate Assignment Expert System. Computer Engineering (6), 69–71 (2000)
8. Tian, C., Nai, G.X.: Genetic-based Airport Gate Assignment Strategy. Computer Engineering 31(3), 186–188 (2005)
9. Wen, J., Li, B., Wang, Q.R., Du, W.: Airport Gate Assignment Graph Coloring Model and Algorithm. Systems Engineering Theory Methodology and Applications 14(2), 136–140 (2005)
10. Luo, R.W., Xie, R.H., Zhang, D.Z.: Airport Gate Assignment Graph Coloring Model and Algorithm. System Engineering Theory and Practice (November 2007)