The scheduling of planes at an airport

By

WEI ZHU

Supervised by Dr. Joab Winkler

Module: Dissertation Project (2016~2017)

This report is submitted in partial fulfilment of the requirement for the degree of MSc Software System and Internet Technology by Wei ZHU

Declaration:

Abstract

It is widely accepted that the Aircraft Scheduling Problem is a critical issue. The arrival delays not only influence people's daily life, but also waste a large number of penalty cost. This thesis mainly focuses on solving the Aircraft Landing Problem, which aims at minimizing the total time cost for landing the aircraft sequence. A dynamic algorithm based on MIP (Mixed Integer Programming) is presented in this report and a program written in JAVA is provided to implement the algorithm. The program tests a set of random data and the results that compared with FCFS rules will be given in discuss section.

Keywords: dynamic programming, mixed integer programming, aircraft scheduling

Acknowledgments

First of all, I would like to thank my superviser Dr. Joab Winkler. He taught me how to search useful information and materials via Google, which helped me a lot when I study on this project. Then I would like to thank my parents who support me to complete my study abroad.

Contents

List of figure

List of Table

Chapter 1

Introduction

1.1General introduction

1.1.1Background

Over the past few decades, air traffic demand has experienced a dramatic growth due to globalization around the world. According to Boeing (2013), the amount of passengers and cargo tonnage are likely to double in the next twenty years. However, most airports are facing aircraft congestions and delay problems. In the report of Ball et al. (2010), approximately one fourth air flights delayed more than 15 min to their destination. According to Cook and Tanner (2011), the total cost of air traffic flow management (ATFM) delay is 1.25 billion euros in 2011.

To reduce the delay costs and solve the aircraft congestion, a large amount of researches have focused on the Aircraft Scheduling Problem (ASP).

ASP has two main parts, the Aircraft Landing Problem (ALP) and the Aircraft Take-off Problem (ATP). For an airport, the amount of aircrafts that land and depart in one hour is the most significant criterial of the measurement of the capacity. For instance, there are approximately 90 aircraft movements in London Heathrow airport per hour in rush time. However, Due to the investment cost, most airports do not have too many runways. Heathrow airport has two runways, one for landing and another one for departing. In London Gatwick airport, there is only one runway for a mixed operation.

Arkind (2004) identified the runway throughput as the bottleneck for ASP because computing an optimal scheduling of plane is a more practical means to reduce the delay rather than other methods, such as building a new runway.

In current aircraft sequences scheduling system, FCFS rule which means first come first serve, is widely used (Brentnall and Chen, 2009). However, according to Dear (1976), FCFS rule cannot give the most optimal solution to make the most throughput of the runway. On the ground of that, many models and algorithms based on different objectives have been proposed by researchers.

1.1.2Aim and objective

According to Sheng Pengyu et al. (2011), the ALS problem is in nature NP-hard, which means that although an optimal algorithm or solution performs well in the model, it may not meet the requirements of the situation in real world. This thesis considers the conditions and constraints in real situations and tries to give more optimal solution to meet the requirements.

The most common objective is to minimise the total cost of delay. However, many researchers also proposed different models and algorithms based on different objectives.

Based on the articles of Idris et al. (1998) and Fahle et al. (2003), the objectives of ALS problem are listed below:

In terms of safety and efficiency:

- Maximizing the runway throughput (total makespan)
- Minimizing the total delay time
- Minimizing the latest time of the landing aircraft

In terms of airline's perspective:

- Minimizing the total fuel cost
- Minimizing the total number of delay passenger
- Ensure the airplanes be as punctual as possible with the timetable

In terms of government's perspective:

- Minimizing the noisy affect
- Minimizing the air pollution affect

For example in detail, Beasley et al. (2000) proposed an algorithm that minimises the deviation from planned arrival time.

The objective of this thesis is to propose an optimal dynamic programming (LP) algorithm aims on minimizing the total time cost of the sequence in a one-runway model for landing aircrafts. Simultaneously, it will also propose the encoding of the aircraft sequencing system with JAVA programming.

Furthermore, some improvement of the algorithm will be discussed and it will try to solve other objectives (minimizing the total cost) based on the algorithm and the software.

1.2 Structure of the thesis

The report has given the background of this project above and states the objectives of this thesis, then several methods and algorithms on APL problem that proposed in the past few years will be discussed in the literature review section. After that, the formulas with related constraints and notations will be presented in requirements and analysis section. In the design section, the algorithm of this project will be explained step by step. Furthermore, this report will give the process of the implementation of the software and compare the testing results with some other algorithms. Finally, some improvement of the algorithm will be stated and some findings and conclusion will be proposed.

Chapter 2

Literature Review

In this section, several models and algorithms on ALP problem will be reviewed and discussed.

2.1 Review of ASP (Aircraft Schedule Problem)

The aircraft schedule problem (ASP) has been researched in the past three decades. Generally, ASP consists of two parts, ATP (aircraft take-off problem) and ALP (aircraft landing problem). This section mainly focuses on the review of ALP.

There are three stages of ALP: sequence the aircrafts, determine the landing time and runway assignment. Mesgarpour et al. (2013) define the three stages as: arrange a schedule, adjust the schedule and finally freezing the schedule. The original order of the sequence is decided by FCFS (first come first serve) rule. If a new aircraft enters the sequence and requires joining in the landing schedule, it should send a signal to the radar 30 minutes before touch down. Neuman and Erzberger (1991) state that the schedule is frozen three minutes before landing the sequence because the aircraft is too close to the airport to re-schedule the sequence. A large number of models and algorithm have been proposed based on different objectives and all the models could be classified into two categories, static case and dynamic case (Beasley et al., 2004).

This chapter will review several models and algorithm next.

2.2 Review of Dynamic Programming

Dynamic programming is a common method to solve sequence problem. It is widely accepted that ALP (aircraft landing problem) could use dynamic programming to schedule the sequence.

The first attempt to address the problem via DP is proposed by Dear (1976). He established a model that each aircraft in the sequence has a constraint that limits the position in the sequence where aircraft could be scheduled. The constraint is called (CPS), which is accepted in other’s article on ALP research. Psaraftis (1980) proposed an application based on Dear’s model to address ALP problem. However, he assumes that all the aircrafts in the sequence could land immediately, which is not realistic in real situation. According to his formulas in the algorithm, the total delay time could be achieved by calculating the sum of the time that each aircraft delayed. Another objective of his algorithm is to maximum the throughput of the runway by calculating the landing time of the last aircraft in the sequence. In terms of one-runway model, the time complexity of the model is $O(C_{n^c})$, where c represents the number of planes in the sequence and C represents the number of flight types in the sequence. An algorithm that based on two-runway model is also introduced in his article.

In 2006, Brentnall improved the algorithm of Psaraftis. He combines more constraints into the considerations. For example, he defines an earliest landing time parameter for each aircraft and he assumes that the aircrafts with less separation time should land first. Based on his assumption, two objectives could be achieved, one is to minimize the total time for landing the sequence and another one is to minimize the landing time of the last plane. Brentnall divided the sequence into K stacks, the plane which is selected to land next will be chosen from the bottom aircraft of the stacks. The time complexity of the algorithm is $O(n^{c^2+k})$, where K means the number of the stacks.

Balakrishnan and Chandran (2010) proposed a dynamic programming frame for ALP. The objective is also to minimize the landing time of last aircraft. Furthermore, they consider more constraints in their algorithm, such as time window and separation time. Based on this model, Lee et al. improved the algorithm to calculate the total fuel cost for landing the sequence because the fuel cost is the main criterion that the airplane company concerns.

A set of data from Dallas-Fort Worth Airport was tested by the algorithm and the conclusion that Lee et al. draw is that maximizing the throughput of the runway is the most optimal objective to improve the performance of aircraft scheduling compared with minimizing the total delay time and minimizing total fuel cost. They also defined a criterion that the solution which saves more than 3 minutes is an optimal solution.

Briskorn et al. (2014) develop a dynamic programming algorithm based on multiple runways model. The constraints of time window are considered but the computational testing result was not given.

In 2016, Bennell et al. proposed a model with multiple objectives. In order to improve the efficiency of the algorithm, they adopt the methods that Lieder suggested in 2014. The experimental results with Heathrow International Airport indicate that the algorithm could give optimal solution both for static cases and dynamic cases.

2.3 Review of Genetic Algorithm

Genetic algorithm (GA) is another common approach to solve sequencing problems, which is mentioned in many reports. To apply the GA to ALP, a chromosome is needed to illustrate the order of the planes. Beasley et al. (2001) defined the landing time of each aircraft as the chromosome.

One of the first attempts to address the ALP with GA is given by Stevens in 1995. The aim of the algorithm is to minimize the total penalty cost for landing the sequence. However, the separation time in that model is a constant, which is 3 minutes. With the constraints and considerations, 40 sets of data are tested and the computational results are given in his report.

Cheng et al. (1999) proposed 4 models based on GA to solve multiple runways ALP. The first GA algorithm encodes the sequencing step with one set of chromosome and encodes the runway assignment step with another set of chromosome. Meanwhile, the chromosome in the second and third model determines the order of the aircrafts in the sequence. However, the fourth model is totally different from the previous three. The fourth model adopts mathematical operations as the chromosomes. The four models test the same set of data and the discussion and evaluation are given in their article. Hansen developed a program to test the efficiency of the GA algorithm and the results indicate that the generic programming could provide optimal solutions successfully.

In 2009, Yu et al. designed a cellular automation (CA) model to solve one-runway scheduling problem. GA algorithm is also used in the model to enhance the performance of the model. They used the data set which provided by Beasley et al. in 2000 and compared the results with Ant Colony algorithm.

In 2005, Hu and Chen proposed a genetic algorithm to deal with dynamic ALP cases. They used a method called receding horizon control (RHC) and compared RHC with other models. Figure 1 shows the details of comparisons.

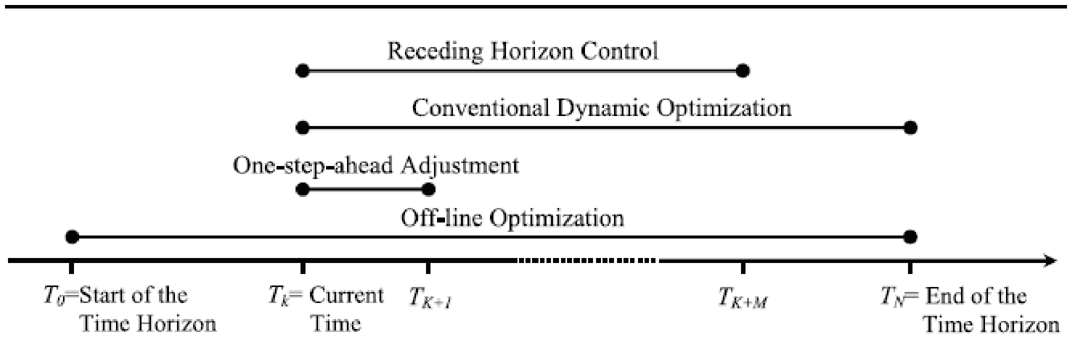

Receding Horizon Control
Conventional Dynamic Optimization
One-step-ahead Adjustment
Off-line Optimization
T_0=Start of the Time Horizon
T_k= Current Time
T_{K+1}
T_{K+M}
T_N= End of the Time Horizon

Figure 1. comparisons between different models (Hu&Chen, 2005)

In figure 1, the off-line optimization aims at minimize total time cost; the one step-ahead adjustment method is to modify the sequence dynamically. In addition, the RHC and conventional dynamic optimization also have different objectives. Hu and Chen also analyze the performance of their model based on GA with different level of uncertainty.

2.4 Branch-and-bound Algorithm

One of the first branch-and-bound algorithms for ALP is proposed in 1992 by Brinton. The algorithm could deal with both static and dynamic cases. The objective of the algorithm is to minimize the combination cost even though the detailed types of cost are not mentioned in the report. The implementation of the algorithm is analyzed even though none of testing results are given.

Abela et al. suggested a branch-and-bound algorithm in 1993, which focuses on one-runway model. The objective is to calculate the delay cost of each aircraft. Meanwhile, they presented the testing results of a set of data with considerations of separation time.

Beasley et al. proposed a branch-and-bound algorithm based on linear programming, which is suitable both for single runway model and multiply runways model. The formula of Ernst et al. is adopted to calculate the total penalty cost. Some constraints are also considered to enhance the efficiency of calculation of mix integer formula. The algorithm was tested with the data of a 50 aircrafts sequence.

2.5 Ant colony optimization

Ant colony optimization (ACO) is not a common method to solve ALP. Randall designed an ACO algorithm to minimize the total penalty cost. Some vital constraints are also considered, such as earliest landing time and latest landing time. However, the results indicate that the solution that the algorithm gives is not as optimal as expected.

In 2009, Bencheikh et al. suggested a hybrid method which is the combination of generic algorithm and ant colony optimization. The performance of the method is better than the generic algorithm. He improved the algorithms in 2011 even though he did not give the detailed formulas. However, the testing results indicate that the algorithm could give optimal sequence in 80% of the cases.

In 2016, Bencheikh et al. made more improvement on the model they proposed before. They modified the formulas in the static cases so that the formulas could meet the requirements of dynamic cases as well. The comparisons between other three algorithms were given by Beasley et al. in 2006.

2.6 Other Algorithms

There are some other algorithms that proposed to solve ALP.

Wen et al. (2005) suggested an algorithm called branch-and-price. They used the objective function which was proposed by Ernst et al. in 1999. A set of data from Beasley (1990) was tested and the results show that the model gives a lower bound in

MIP model.

Another method called queuing theory is proposed by Bauerle et al. in 2007. The theory defines an M/SM/1 queuing system based on two runways model. The average delay time that the algorithm gives is compared with other algorithm in the report.

Chapter 3

Requirements and analysis

3.1 Basic concepts

The aim of Air Traffic Management (ATM) system is to ensure the airlines depart and land safely and efficiently. There are three main facilities in ATM system, traffic control tower, terminal airspace control centre and en-route control centre. Neufville and Odoni (2003) gave details about different roles the three facilities play in ATM.

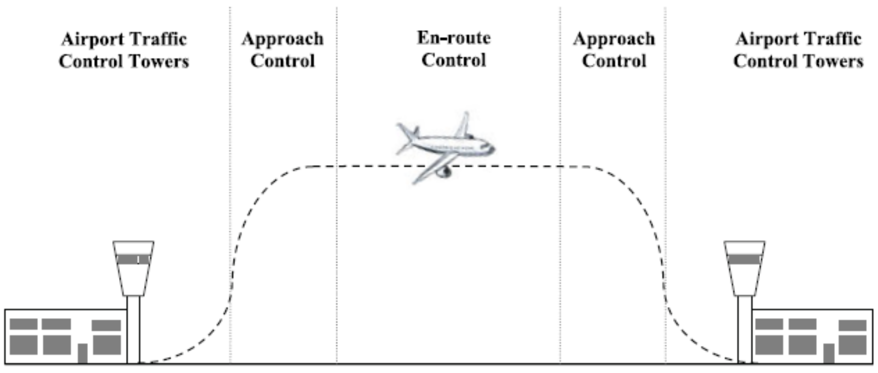

Airport Traffic Control Towers
Approach Control
En-route Control
Approach Control
Airport Traffic Control Towers

Figure 2 Air traffic control segments (Bennell, 2013)

When an aircraft is approaching the airport, the control tower will receive a signal from the aircraft when it entre the range of radar at the airport. The signal will transmit the relevant information of the aircraft, such as flight number, speed and altitude. Accordingly, the control tower will give the aircraft an approaching corridor and required speed.

3.2 Problem description

Sheng Pengyu et al. (2011) state that scheduling aircraft sequences problem could be divided into two main parts. The first part is to determine the order of the aircrafts in the sequence, the second part is to assign an exact landing time for each aircraft with constraints. In this project, I assume that all the flights are assigned to land in the same runway. At the same time, all the runways are independent, which means that the runways are separated sufficiently.

3.3 Design and model

3.3.1 Assumptions

In this model, I assume that there is a given set of aircrafts, A = {1, 2……N}, N∈ R^+, each aircraft has a type, small, large or heavy. Each aircraft has an earliest landing time T_e, latest landing time, T_l and an estimate landing time T_t. All the T_e, T_l and T_t will be given when the sequence has been initialized. All the aircrafts are assigned to land in a single runway.

3.3.2 Constraints

Time window:

For each aircraft, there should be a time window $[T_e, T_l]$, and the exact landing time T_i that control tower determines should lie within the time window.

$$T_{ei} < T_i < T_{li} \qquad \forall\, i \in R^+ \qquad (1)$$

Normally, the earliest landing time is obtained in the case that the aircraft flies to the destination at its maximum safe speed. The latest landing time typically decided by the fuel carried on the aircraft.

Minimum Separation time:

A wake-vortex (WV) will be created when an airplane flies, the WV will affect the status of the following airplanes. To ensure the stability and safety of the following airplanes, the Federal Aviation Administration (FAA) classifies the airplanes into three classes: small, large, heavy. The specific separation time between each type is listed in the following table:

	The following aircraft			
The Leading aircraft		Small	Large	Heavy
	Small	82	69	60
	Large	131	69	60
	Heavy	196	157	96

Table1. The minimum separation time between each type of aircraft in seconds (Ma Weimin et al., 2014)

There is one point that needs to be mentioned, the WV that a large aircraft generates is much stronger than that a small aircraft generates. As a result, the separation time of the same sequence may be different if the order is changed. Figure 3 illustrates this situation.

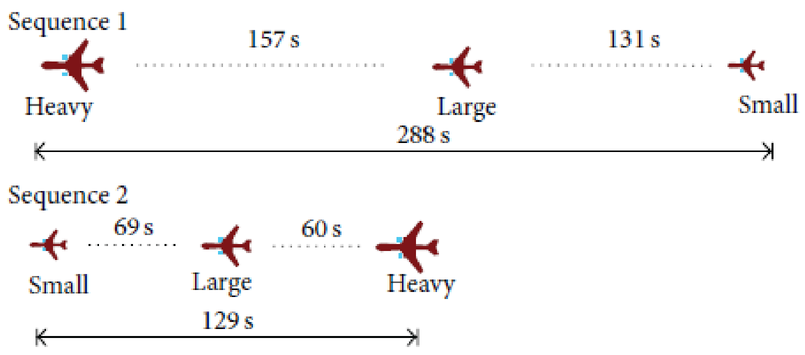


Figure 3. Different separation time of different order sequence (Ma Weimin et al., 2014)

Chapter 4

Design

4.1 Model and formulation

In this section, a MIP (Mixed Integer Programming) model is proposed. The MIP model of ALP problem is based on the model which Alexander et al. gave in 2014.

The notations of the model are listed in the following table:

A	The set of aircrafts
W	Number of flight types(in this case, W=3)
R	Number of runways(in this case, R=1)
w(a)	Type of aircraft $a \in A: w(a) \in \{1, \dots \dots, W\}$
T_a	Target landing time (or estimate landing time) of aircraft a

L_a	Latest landing time of aircraft a
$S_{w,w'}$	Separation time between aircraft w and w’
M	A sufficiently large number
C_a	The determined landing time of aircraft a
$\gamma_{a,a'}$	$\begin{cases} 1 & if\ a\ lands\ after\ a' \\ 0 & otherwise \end{cases}$
f_{ar}	$\begin{cases} 1 & if\ a\ lands\ first\ of\ the\ sequence \\ 0 & otherwise \end{cases}$
l_{ar}	$\begin{cases} 1 & if\ a\ lands\ last\ of\ the\ sequence \\ 0 & otherwise \end{cases}$

Table 2. The notations of the MIP model (Alexander et al., 2014)

In the Model of Alexander et al., they assume that

For each aircraft a, the determined landing time must lie in the time window, in the form of formula is that:

$$T_a \leq t \leq L_a \tag{2}$$

For each pair of aircraft, the minimum separation time must be satisfied, in the form of formula is that:

$$T_j - T_i \geq S_{w(i),w(j)} \tag{3}$$

If any solution satisfies the above two conditions, this solution is called feasible solution. Besides that, the objective of their model is to minimize the total delay cost, and the formula is:

$$\sum_{(a,r,t)\in S} C_{w(a)}(t - T_a) \tag{4}$$

However, in this thesis, in order to make the model closer to the situation in real life, I consider all the possible solutions and check each solution in the algorithm which is not feasible defined in their article.

The objective of this thesis is to minimize the total time of landing the sequence, so the formula (4) should be modified as following:

$$\sum_{(a,r,t)\in S}(t - T_a) \tag{5}$$

The formula (5) should subject to the following constraints:

$$T_a \le C_a \le L_a \qquad \forall \mathrm{a} \in \mathrm{A} \tag{6}$$

$$C_a + S_{w(a)w(a')} \le C_{a'} + M(1 - \gamma_{aa'}) \quad \forall a, a' \in A; a \ne a' \tag{7}$$

$$C_a \le C_{a'} \quad \forall a, a' \in A;\ T_a < T_{a'}; w(a) = w(a') \tag{8}$$

$$\sum_{a'\in A} \gamma_{a'a} + \sum_{r=1}^{R} f_{ar} = 1 \ \ \forall a \in A \tag{9}$$

$$\sum_{a'\in A} \gamma_{a'a} + \sum_{r=1}^{R} l_{ar} = 1 \ \ \forall a \in A \tag{10}$$

$$\sum_{a\in A} f_{ar} \le 1 \quad \forall r = 1, \dots, R \tag{11}$$

$$\sum_{a\in A} l_{ar} \le 1 \quad \forall r = 1, \dots, R \tag{12}$$

$$C_a \ge 0 \quad \forall a \in A \tag{13}$$

$$\gamma_{aa'}, f_{ar}, l_{ar} \in \{0; 1\} \ \ \forall a, a' \in A\ ,\ \forall r = 1, \dots, R \tag{14}$$

To assign the exact landing time of each aircraft, Balakrishnan and Chandran (2006) proposed a dynamic programming algorithm and the recursion formula is:

$$T_j = max\{e(i), \min_{i \in P(j): T_i \leq l_i}\{T(i) + s_{ij}\}\} \tag{15}$$

The objective of this formula is to minimize the exact landing time of each aircraft, which is closer to this project's aim. As a result, formula (15) will be adopted in the implementation section to determine the exact landing time of each aircraft.

4.2 Algorithms

4.2.1 Review of DP (Dynamic programming)

In this project, I combine the MIP model and the dynamic programming proposed by Balakrishnan and Chandran (2006).

Dynamic Programming is a common optimization method to solve sequential problems. In 1978, Psaraftis proposed a model that each aircraft in the sequence could land immediately. His DP algorithm and formula aim to maximize the throughput of the runway by calculating the sum of the landing time. In 2006, based on the algorithm of Psaraftis, Brentnall established an improved DP algorithm. He assumes that each aircraft is sequenced in a non-decreasing order. Subject to this assumes the algorithm could both minimize the last aircraft's landing time and minimize the total landing time. Balakrishnan and Chandran (2006) improved the algorithm and the objective is also to minimize the last aircraft's landing time. However, more constraints about time window are considered in the algorithm, which makes the algorithm closer to the situation in real life. The algorithm transforms the Landing sequence problem into shortest path problem. Figure 4 indicates the process of finding all possible position of the aircraft in the sequence.

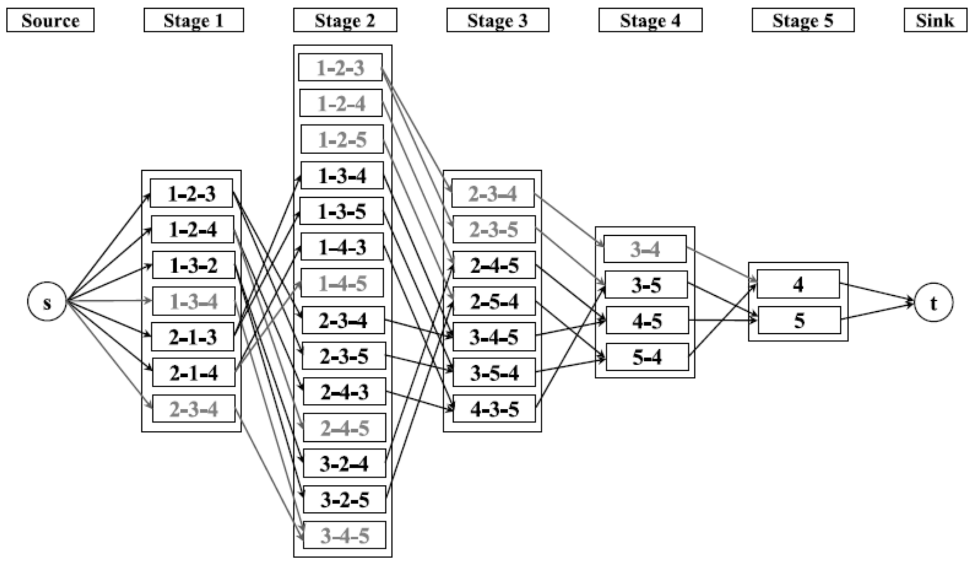


Figure 4. Process of the network where n=5, k=1 (Balakrishnan and Chandran, 2006)

The algorithm uses a dynamic programming recursion method to determine the exact landing time of each aircraft, which is shown below:

$$T_j = max\left\{e(i), \min_{i\in P(j):T_i\leq l_i}\{T(i)+s_{ij}\}\right\}$$

A set of realistic data of Denver International Airport was tested and the results showed optimal solutions.

4.2.2 Algorithm Design

In this project, the algorithm is divided into three steps.

Step 1: use a dynamic programming recursion method to calculate all the possible order of the sequence

Step 2: use formula (15) to determine an exact landing time for each aircraft in the sequence

Step 3: check whether the order of the sequence with minimum total time is feasible ($T_a \leq t \leq L_a$ and $T_j - T_i \geq S_{w(i),w(j)}$)

Step 1: Order Permutation

In order to find all the possible orders of the sequence, I use a recursion algorithm which similar with the algorithm of Balakrishnan and Chandran. The aim of the algorithm is to swap positions of all elements.

In the first stage, the position of element A in the set will be immobilized and then swap all positions of the subset except A.

In order to swap all positions of the subset, I recall the recursion method and input the subset except A as a new set. Then the position of element B in the subset will be immobilized and swap all position of the subset except element B.

Do the loop until there are only two elements in the set.

Finally, when the length of the processed set equals the length of the original set, all the orders of the set will be listed.

Pseudo-Code of step 1:

```
Input: aircraft set (a1, a2… an), input the parameters of each aircraft (w (a), $T_a$, $L_a$) via the interface

Output: all the orders of the aircraft sequence

Initialize the sequence of the aircraft set

If the length of input set A' = length of original set A

    Add input A' into result set R

  Else:

    For every subset B of original set A

        Swap elements of the B

Repeat:
```

Recall the recursion method

End

Step 2: Determine Landing Time

In order to determine the exact time of each aircraft in the sequence, we need to calculate the separation time between each pair of the aircraft.

According to table1, we can easily find separation time between different types. Then input the values of constraints in formula (15), we could obtain the exact landing time of each aircraft.

Step 3: Feasibility Check

As long as we determine the landing time for each aircraft, we could calculate the total time of each order of sequence via formula (5): $\sum_{(a,r,t)\in S}(t - T_a)$. Eventually, we need to check whether the minimum result is feasible via formula (2): $T_a \le t \le L_a$ and formula (3) $T_j - T_i \ge S_{w(i),w(j)}$ and find the feasible minimum solution.

The flow char of the algorithm is shown below:

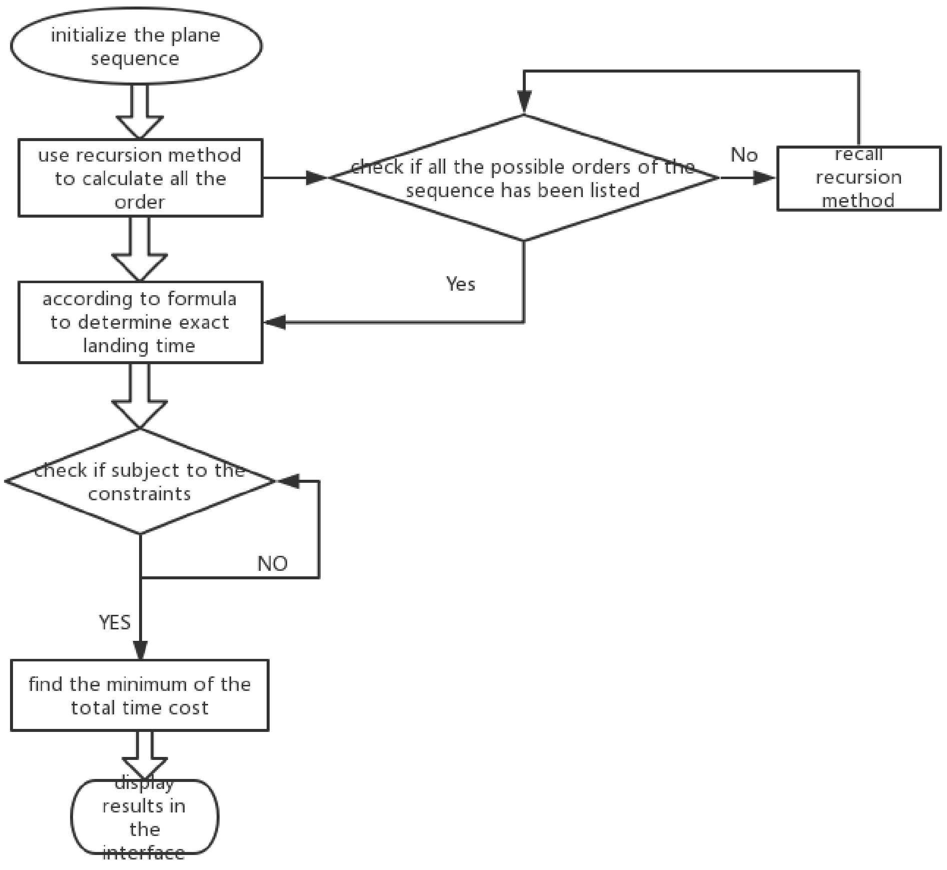

initialize the plane sequence
use recursion method to calculate all the order
check if all the possible orders of the sequence has been listed
No
recall recursion method
Yes
according to formula to determine exact landing time
check if subject to the constraints
NO
YES
find the minimum of the total time cost
display results in the interface

Figure 5. The flow chart of the algorithm

4.3 Program design

In this project, I choose JAVA as my programming language. The interface is designed based on the Swing package in JAVA JDK 1.8.

The program aims to provide a simple interface to input the parameters of each aircraft and display the solutions. On the ground of that, the user story and functional story is not complex.

User story

- User could input the number of each plane

- User could input the earliest landing time of each plane
- User could input the latest landing time of each plane
- User could input the estimate (target landing time) landing time of each plane
- User could select the type of each plane
- User could add new plane into the sequence
- User could delete any plane in the sequence

Functional story

- The program could add a new plane with parameters into the sequence after click ‘Add aircraft’ button
- The program could display the new sequence in the left table after click ‘Add aircraft’ button
- The program could delete any plane with parameters in the sequence after click ‘Delete aircraft’ button
- The program could display the new sequence in the left table after click ‘Delete aircraft’ button
- The program could run the algorithm successfully
- The program could display the process of each step of the algorithm in the console
- The program could display the feasible minimum total time for landing the sequence
- The program could display the optimal order of aircraft sequence and the parameters of each aircraft in the right table

According to the user story and functional story, the structure of the aircraft class could be defined. It should have a certain number of parameters: flight No., earliest landing time, latest landing time, estimate landing time, separation time and flight type. Some methods should also be defined to satisfy the program. The UML chart of aircraft class is shown below:

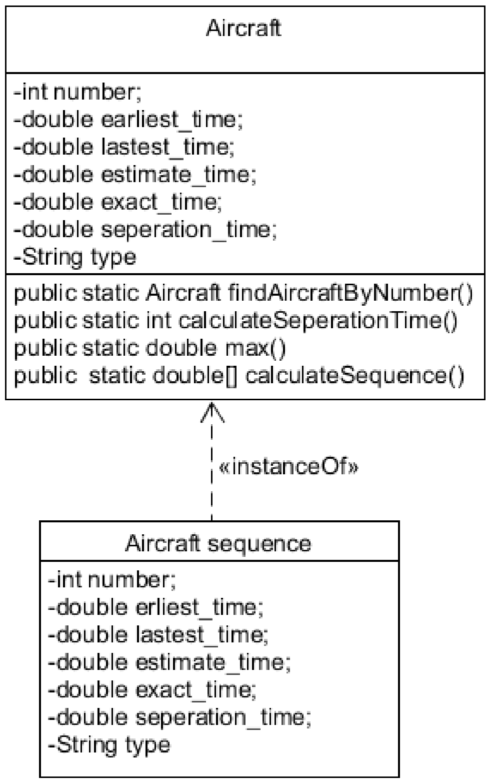


Figure 6. The UML of the aircraft class

Chapter 5

Implementation and testing

5.1 Interface implementation

In this project, I choose JAVA as my programming language. The interface is designed based on the Swing package in JAVA JDK 1.8.

Swing is a foundation class of JAVA for design graphic user interface (GUI). It is more powerful than AWT (Abstract Window Toolkit).

The interface is shown below:

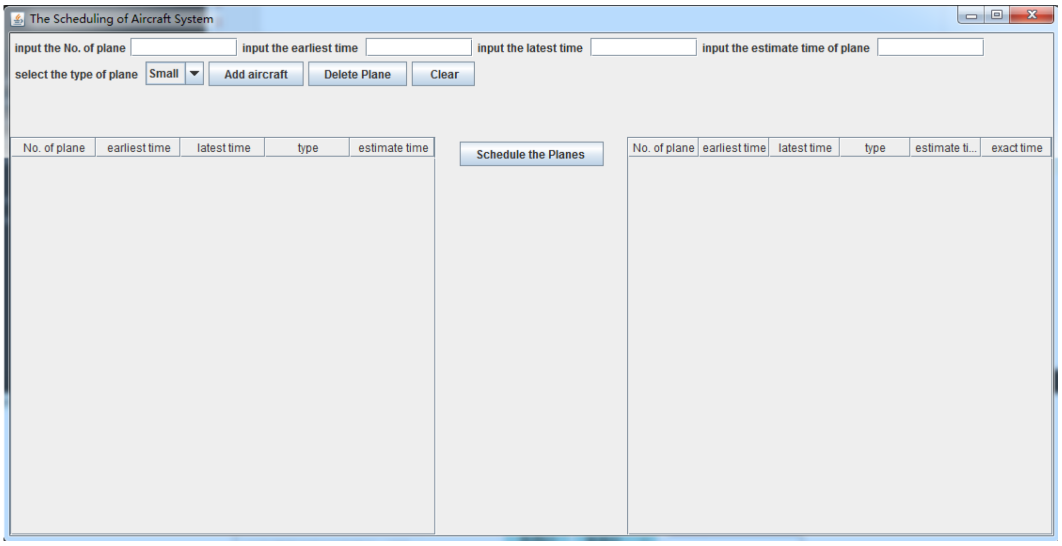
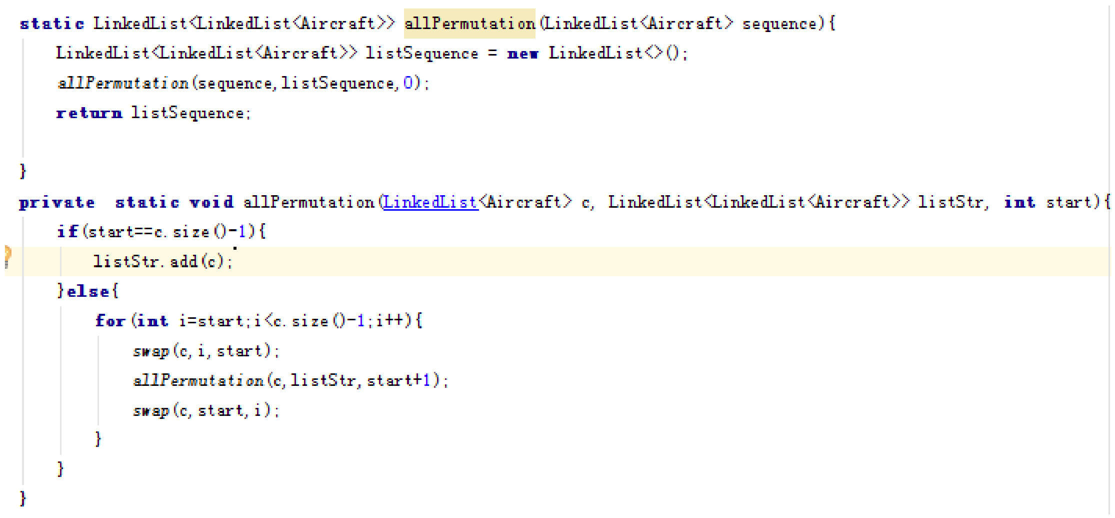


Figure 7. The interface of the program

5.2 Algorithm implementation

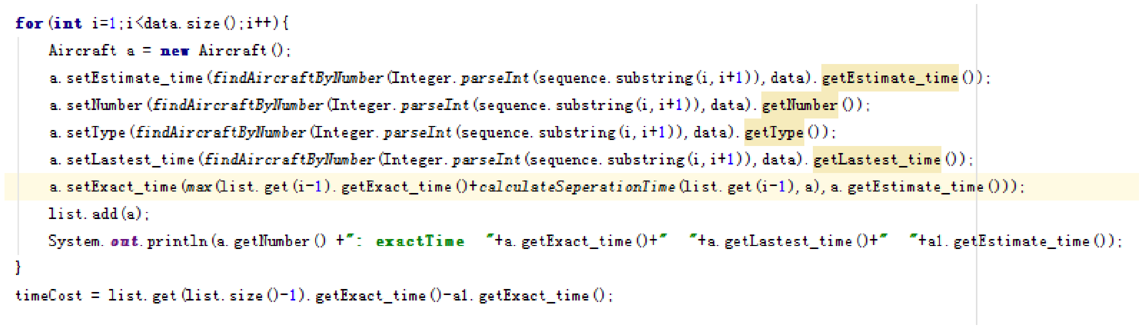
The core code of implementation of each step is shown below: Figure 8 illustrates the recursion method of step 1; Figure 9 illustrates the main process of step 2 and step 3.



```java
static LinkedList<LinkedList<Aircraft>> allPermutation(LinkedList<Aircraft> sequence){
    LinkedList<LinkedList<Aircraft>> listSequence = new LinkedList<>();
    allPermutation(sequence,listSequence,0);
    return listSequence;

}
private  static void allPermutation(LinkedList<Aircraft> c, LinkedList<LinkedList<Aircraft>> listStr, int start){
    if(start==c.size()-1){
        listStr.add(c);
    }else{
        for(int i=start;i<c.size()-1;i++){
            swap(c,i,start);
            allPermutation(c,listStr,start+1);
            swap(c,start,i);
        }
    }
}
```

Figure 8. Recursion method to find all possible order in sequence



```java
for(int i=1;i<data.size();i++){
    Aircraft a = new Aircraft();
    a.setEstimate_time(findAircraftByNumber(Integer.parseInt(sequence.substring(i,i+1)),data).getEstimate_time());
    a.setNumber(findAircraftByNumber(Integer.parseInt(sequence.substring(i,i+1)),data).getNumber());
    a.setType(findAircraftByNumber(Integer.parseInt(sequence.substring(i,i+1)),data).getType());
    a.setLastest_time(findAircraftByNumber(Integer.parseInt(sequence.substring(i,i+1)),data).getLastest_time());
    a.setExact_time(max(list.get(i-1).getExact_time()+calculateSeperationTime(list.get(i-1),a),a.getEstimate_time()));
    list.add(a);
    System.out.println(a.getNumber() +": exactTime  "+a.getExact_time()+"  "+a.getLastest_time()+"  "+a1.getEstimate_time());
}
timeCost = list.get(list.size()-1).getExact_time()-a1.getExact_time();
```

Figure 9. Use formula (15) to determine the exact landing time

5.3 Testing

5.3.1 Testing data definition

In order to test the program, firstly we need to define a sequence of aircraft with some certain constraints and considerations.

Flight No.	Earliest landing time	Latest landing time	Estimate landing time	Type
1	0	1000	23	small
2	0	1000	35	Small
3	0	1000	76	Small
4	0	1000	211	Small
5	0	1000	142	Small
6	0	1000	344	Large
7	0	1000	113	small
8	0	1000	411	heavy

Table 3. The definition of testing data

Some considerations are also defined as following:

All the aircrafts in the sequence are assigned to land in a single runway

All the aircrafts in the sequence land in the same direction

There is satisfied minimum separation distance between each pair of aircrafts

5.3.2 Testing results

In this section, each user story and functional story in design section has been tested, all the stories passed the test. The user could add new aircrafts into the sequence or delete aircrafts in the sequence. Accordingly, the results will also be changed and displayed in the interface. The feasibility of the solutions will be discussed in the next section.

Chapter 6

Results and discussion

6.1 Analysis

In this experiment, the settings of test computer are 2.7 GHz Intel Core i7 processor, 8 GBAM.

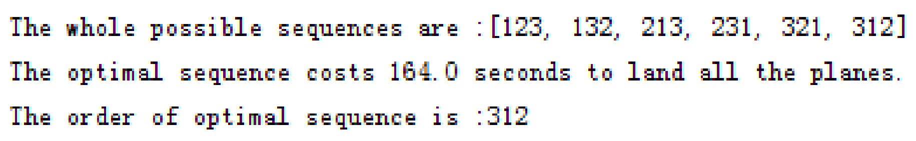
In order to check whether the program will give the stable output with the same input values, the same data in the previous section has been tested 5 times and the results are given in table 4.

Flight No.	**Earliest landing time(s)**	**Latest landing time(s)**	**Estimate landing time(s)**	**Flight type**	**Determined landing time(s)**
5	0	300	142	small	142
1	0	1000	23	small	224
2	0	1000	35	small	306

3	0	1000	76	small	388
4	0	1000	211	small	470
7	0	1000	113	small	552
6	0	1000	344	large	621
8	0	1000	411	heavy	681
Total time cost	539				

Table 4. Testing results

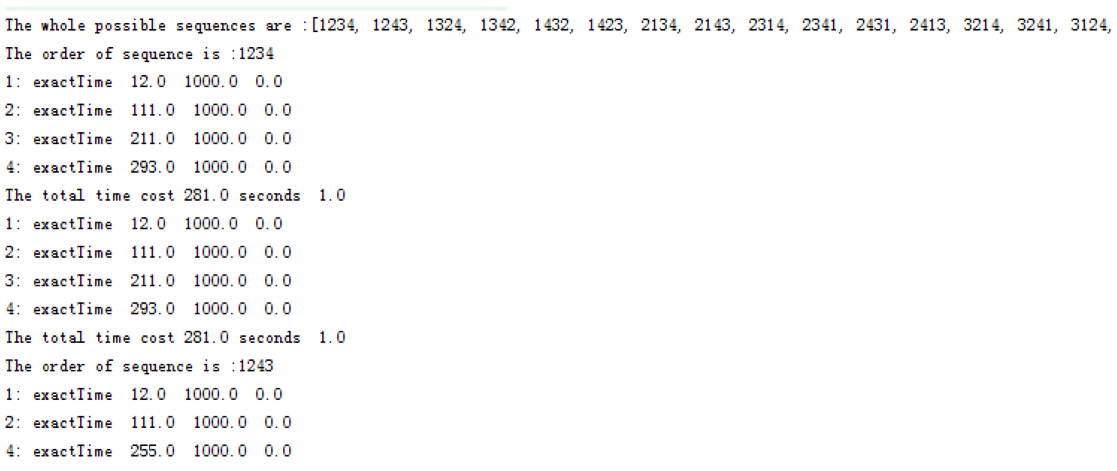
Firstly I input aircraft No. 1 to No. 3 in the sequence because there will be too much output of all possible orders if the size of input set is too large. The results of algorithm are shown below:



```
The whole possible sequences are :[123, 132, 213, 231, 321, 312]
The optimal sequence costs 164.0 seconds to land all the planes.
The order of optimal sequence is :312
```

Figure 10. The results of algorithms with a part of input

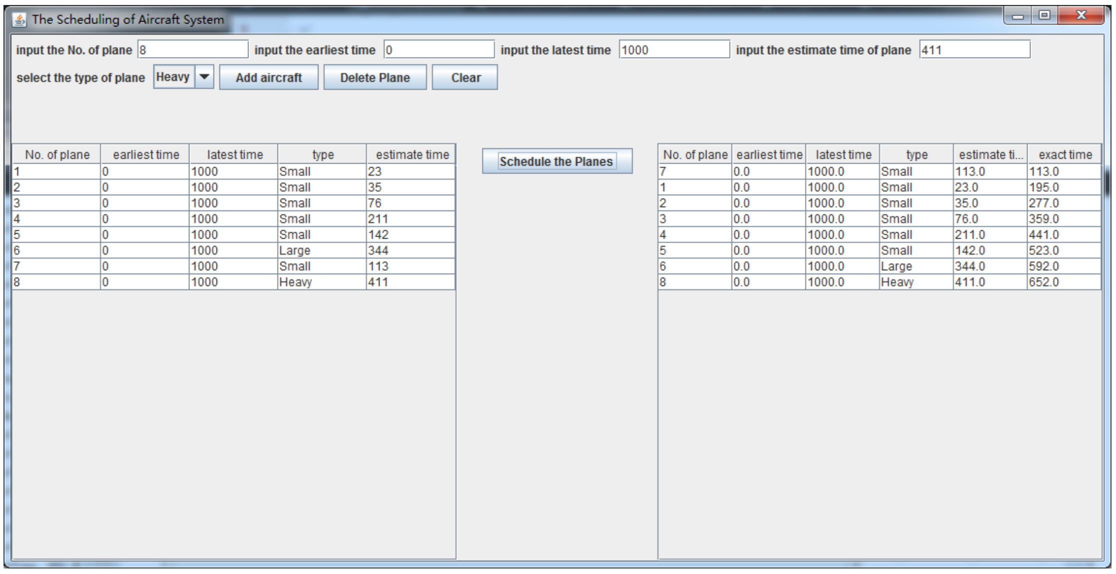
In order to test step 2 and step 3 of the algorithm, I input more data to the sequence. Figure 11 indicates a part of the process of the algorithm.



```
The whole possible sequences are :[1234, 1243, 1324, 1342, 1432, 1423, 2134, 2143, 2314, 2341, 2431, 2413, 3214, 3241, 3124,
The order of sequence is :1234
1: exactTime  12.0  1000.0  0.0
2: exactTime  111.0  1000.0  0.0
3: exactTime  211.0  1000.0  0.0
4: exactTime  293.0  1000.0  0.0
The total time cost 281.0 seconds  1.0
1: exactTime  12.0  1000.0  0.0
2: exactTime  111.0  1000.0  0.0
3: exactTime  211.0  1000.0  0.0
4: exactTime  293.0  1000.0  0.0
The total time cost 281.0 seconds  1.0
The order of sequence is :1243
1: exactTime  12.0  1000.0  0.0
2: exactTime  111.0  1000.0  0.0
4: exactTime  255.0  1000.0  0.0
```

Figure 11. a part of the process

The optimal solution of the sequence is listed in the right table of the interface and the total time cost will be shown in a new window, which is shown in figure 12 and figure 13.



The Scheduling of Aircraft System

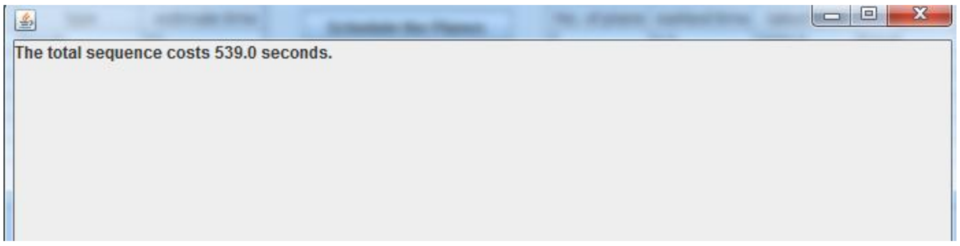
input the No. of plane: 8 | input the earliest time: 0 | input the latest time: 1000 | input the estimate time of plane: 411

select the type of plane: Heavy

No. of plane	earliest time	latest time	type	estimate time
1	0	1000	Small	23
2	0	1000	Small	35
3	0	1000	Small	76
4	0	1000	Small	211
5	0	1000	Small	142
6	0	1000	Large	344
7	0	1000	Small	113
8	0	1000	Heavy	411

No. of plane	earliest time	latest time	type	estimate ti...	exact time
7	0.0	1000.0	Small	113.0	113.0
1	0.0	1000.0	Small	23.0	195.0
2	0.0	1000.0	Small	35.0	277.0
3	0.0	1000.0	Small	76.0	359.0
4	0.0	1000.0	Small	211.0	441.0
5	0.0	1000.0	Small	142.0	523.0
6	0.0	1000.0	Large	344.0	592.0
8	0.0	1000.0	Heavy	411.0	652.0

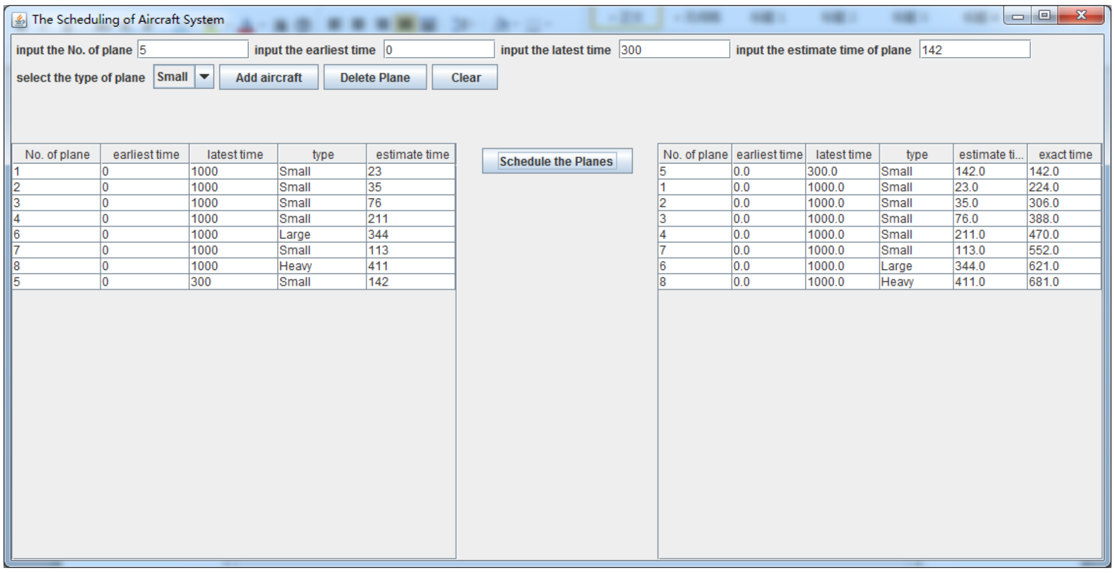
Figure 12. The results of optimal sequence



The total sequence costs 539.0 seconds.

Figure 13. The window to show total time cost

In order to test the function in the step 3 of the algorithm, which is to check whether the solution is feasible ($T_a \leq t \leq L_a$ and $T_j - T_i \geq S_{w(i),w(j)}$), I defined that the latest landing time of flight 5 in the sequence is 300, which means aircraft 5 must land before 300. As a result, the previous solution is not feasible. The new results are shown in Figure 14 and Figure 15after the parameters changed.

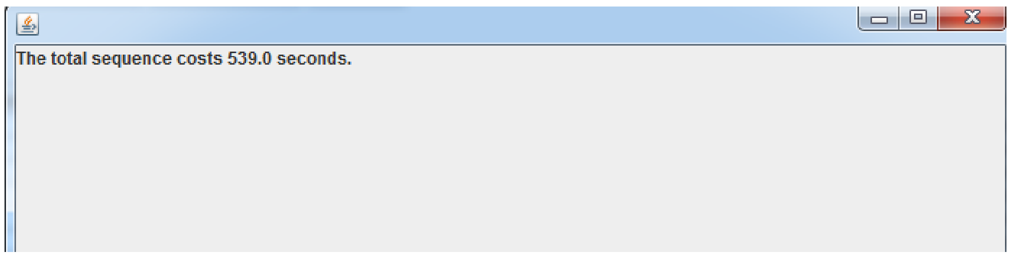


input the No. of plane: 5
input the earliest time: 0
input the latest time: 300
input the estimate time of plane: 142
select the type of plane: Small

No. of plane	earliest time	latest time	type	estimate time
1	0	1000	Small	23
2	0	1000	Small	35
3	0	1000	Small	76
4	0	1000	Small	211
6	0	1000	Large	344
7	0	1000	Small	113
8	0	1000	Heavy	411
5	0	300	Small	142

No. of plane	earliest time	latest time	type	estimate ti...	exact time
5	0.0	300.0	Small	142.0	142.0
1	0.0	1000.0	Small	23.0	224.0
2	0.0	1000.0	Small	35.0	306.0
3	0.0	1000.0	Small	76.0	388.0
4	0.0	1000.0	Small	211.0	470.0
7	0.0	1000.0	Small	113.0	552.0
6	0.0	1000.0	Large	344.0	621.0
8	0.0	1000.0	Heavy	411.0	681.0

Figure 14. The results after parameters changed



```
The total sequence costs 539.0 seconds.
```

Figure 15. The total time cost after parameters changed

Figure 10 and Figure 11 give a new optimal order of sequence because the previous order is not feasible after modifying the parameter. The results indicate that the step 3 of the algorithm passes the test. The solution that the program gave is feasible, which meets formula (2): $T_a \leq t \leq L_a$ and formula (3): $T_j - T_i \geq S_{w(i),w(j)}$.

The total time cost of the solution is 539s. As mentioned in the previous section, currently in most cases, FCFS rule is the common method to solve ALP schedule problem. The same data is tested by FCFS rule and the total time cost is 588s. Compared with FCFS rule, the program shows a more optimal solution, which saves approximately 8.3% of total landing time.

6.2 Improvement

As I discussed in chapter 4, in step 2 of the algorithm, we could determine the exact landing time by formula (15). The aim of this thesis is to calculate the total time for landing the sequence. However, based on the algorithm, we could also calculate the total delay cost of the process.

According to Sheng Pengyu et al. (2011), the formula for calculating total cost is:

$$Zi = \begin{cases} g_i(T_i - X_i) & if\ X_i < T_i \\ h_i(X_i - T_i) & if\ T_i < X_i \end{cases} \qquad (16)$$

In formula (16), g_i means the penalty per unit time if the aircraft i land before estimate landing time; h_i means the penalty per unit time if the aircraft i land after estimate landing time.

After we get the penalty of each aircraft, we could use formula (4):

$\sum_{(a,r,t)\in S} C_{w(a)}(t - T_a)$ to calculate the total penalty cost. Eventually, we could find the order of sequence which has the minimum penalty cost.

6.3 Goals achieved and drawbacks

The results of the program and algorithm prove that this thesis proposes a feasible method to schedule the sequence of airplanes at an airport, which aims to minimize the total time for landing all the sequence. The program considers a certain number of considerations which make the model closer to the realistic situations.

However, an important criterion for an algorithm or program is efficiency. The main drawback of the program is that, although the program could give optimal solution, if the user adds more and more aircraft into the sequence, the calculation time of the program will increase linearly.

6.4 Future work

Although the main goal of the project has been achieved, which is to provide a program to schedule the planes at an airport, there is still some further work that could improve the program.

- This thesis is based on one-runway model, the algorithm should be modified if the requirement is multi-runway model.

- More data from the airports in real life should be tested to ensure the reliability of the program.

- More considerations of the ALP (aircraft landing problem) should be considered. For example, the planes of different types should land in different gates.

Chapter 7

Conclusion

This thesis proposes a dynamic algorithm and a program to solve Aircraft Scheduling Problem (ASP), mainly focuses on Aircraft Landing Problem (ALP). The program aims at minimizing the total landing time of the sequence with some realistic considerations and constraints. Meanwhile, the method for calculating total penalty cost is also provided in chapter 7. However, the reliability of the program should be tested further with more data from the airports.

The formula that this thesis adopted is modified based on the model of Alexander et al. (2014), which aims at minimizing the total cost. The algorithm divides the problem into three steps, the details of each step is given in chapter 4. In addition, chapter 4 also explains the process of the program.

In chapter 5, each user story and functional story defined in chapter 4 has been tested, the results show that the program is able to run successfully and displays the results on the interface clearly.

Chapter 6 analyses the results of the solution and compares with other methods. The results show that the program could help to enhance the throughput of runways, however, the efficiency of the program needs to be improved. Drawbacks and further work are also given to enhance the performance of the program.

To summarize, an algorithm based on dynamic programming has been provided in this project. A program written in JAVA has implemented the algorithm. The main objectives of this thesis have been achieved although there are still some disadvantages and drawbacks.

Reference

[1] Warner, M., 2013. Boeing: Current market outlook 2013–2032. *Boeing Commercial Airplanes, Seattle, WA. Google Scholar*.

[2] Ball, M., Barnhart, C., Dresner, M., Hansen, M., Neels, K., Odoni, A., Peterson, E., Sherry, L., Trani, A.A. and Zou, B., 2010. Total delay impact study: a comprehensive assessment of the costs and impacts of flight delay in the United States.

[3] Cook, A.J. and Tanner, G., 2011. European airline delay cost reference values.

[4] Arkind, K.D., 2004. Requirements for a novel terminal area capacity enhancement concept in 2022. *AIAA Paper*, *5411*.

[5] Brentnall, A.R. and Cheng, R.C., 2009. Some effects of aircraft arrival sequence algorithms. *Journal of the Operational Research Society*, *60*(7), pp.962-972.

[6] Dear, R.G., 1976. *The dynamic scheduling of aircraft in the near terminal area*. Cambridge, Mass.: Flight Transportation Laboratory, Massachusetts Institute of Technology,[1976].

[7] Yu, S.P., Cao, X.B. and Zhang, J., 2011. A real-time schedule method for Aircraft Landing Scheduling problem based on Cellular Automation. *Applied Soft Computing*, *11*(4), pp.3485-3493.

[8] Fahle, T., Feldmann, R., Götz, S., Grothklags, S. and Monien, B., 2003. The aircraft sequencing problem. In *Computer science in perspective* (pp. 152-166). Springer Berlin Heidelberg.

[9] Beasley, J.E., Krishnamoorthy, M., Sharaiha, Y.M. and Abramson, D., 2000. Scheduling aircraft landings—the static case. *Transportation science*, *34*(2), pp.180-197.

[10] Bennell, J.A., Mesgarpour, M. and Potts, C.N., 2013. Airport runway scheduling. *Annals of Operations Research*, *204*(1), pp.249-270.

[11] Carr, G.C., Erzberger, H. and Neuman, F., 1998, December. Airline arrival prioritization in sequencing and scheduling. In *2nd USA/Europe Air Traffic Management R&D Seminar* (pp. 34-40). Orlando, Florida, USA: EUROCONTROL and FAA.

[12] Psaraftis, H.N., 1980. A dynamic programming approach for sequencing groups of identical jobs. *Operations Research*, *28*(6), pp.1347-1359.

[13] Balakrishnan, H. and Chandran, B., 2006, August. Scheduling aircraft landings under constrained position shifting. In *AIAA Guidance, Navigation, and Control Conference and Exhibit, Keystone, CO*.

[14] Lieder, A., Briskorn, D. and Stolletz, R., 2015. A dynamic programming approach for the aircraft landing problem with aircraft classes. *European Journal of Operational Research*, *243*(1), pp.61-69.

[15] Hu, X.B. and Chen, W.H., 2005. Receding horizon control for aircraft arrival sequencing and scheduling. *IEEE Transactions on Intelligent Transportation Systems*, *6*(2), pp.189-197.

[16] Brinton, C.R., 1992, October. An implicit enumeration algorithm for arrival aircraft. In *Digital Avionics Systems Conference, 1992. Proceedings., IEEE/AIAA 11th* (pp. 268-274). IEEE.

[17] Abela, J.D.M.A.G., Abramson, D., Krishnamoorthy, M., De Silva, A. and Mills, G., 1993, July. Computing optimal schedules for landing aircraft. In *proceedings of the 12th national conference of the Australian Society for Operations Research, Adelaide* (pp. 71-90).

[18] Ernst, A.T., Krishnamoorthy, M. and Storer, R.H., 1999. Heuristic and exact algorithms for scheduling aircraft landings. *Networks*, *34*(3), pp.229-241.

[19] Randall, M., 2002. Scheduling aircraft landings using ant colony optimisation. In *6th IASTED International Conference Artificial Intelligence and Soft Computing, Banff, Alberta, Canada* (pp. 129-133).

[20] Wen, M., 2005. *Algorithms of scheduling aircraft landing problem* (Master's thesis, Technical University of Denmark, DTU, DK-2800 Kgs. Lyngby, Denmark).

[21] Bauerle, N., Engelhardt-Funke, O. and Kolonko, M., 2003. *On the waiting time of arriving aircrafts and the capacity of airports with two runways*. Working paper available from the first author at Inst. for Mathematical Stochastics, University of Hannover, Welfengarten 1, D-30167 Hannover, Germany, 2003. Currently available from:< http://www. math. tu-clausthal. de/Arbeitsgruppen/Stochastische-Optimierung/VeroeffKLetzt/VeroeffKLetzt. html.

[22] Bianco, L., Dell'Olmo, P. and Giordani, S., 1997. Scheduling models and algorithms for TMA traffic management. *Modelling and simulation in air traffic management*, pp.139-167.

[23] Ma, W., Xu, B., Liu, M. and Huang, H., 2014. An efficient approximation algorithm for aircraft arrival sequencing and scheduling problem. *Mathematical Problems in Engineering*, *2014*.

[24] Idris, H., Delcaire, B., Anagnostakis, I., Hall, W., Pujet, N., Feron, E., Hansman, R., Clarke, J.P. and Odoni, A., 1998. Identification of flow constraint and control points in departure operations at airport systems. In *Guidance, Navigation, and Control Conference and Exhibit* (p. 4291).