Programming Ideas And Steps

An Xu

TFMP

1. Understanding of formulas

This study expands the original TFMP model in order to make the model consider more factors and more complete so that it can be used in more situations. And from the perspective of the practical application, some of the constraints are improved.

First, it is considered that the actual allocation of aircraft departure and arrive by different airports is not fixed, which is a change due to the different allocation schemes of the runway. Therefore, the service capacity of the airport is changing. The original airport capacity limit formula has now been rewritten to include all possible aircraft arrival and departure service constraints. Get the following constraint formula:

$$\alpha_{kt}^{i} \sum_{f:t\in T_f^k, k=P(f,1)} (w_{ft}^{k} - w_{f,t-1}^{k}) + \beta_{kt}^{i} \sum_{f:t\in T_f^k, k=P(f,N_f)} (w_{ft}^{k} - w_{f,t-1}^{k}) \le \gamma_{kt}^{i}$$

Second, consider the connection between flights. It may happen that the plane arrives at the airport, and after a period of time, it will continue to take the next flight, so it is necessary to consider the connection constraints between the flights. A new 0-1 decision variable $x_{f'f}$ has been added here to control the connection of flights. The original flight constraints become in the form of the following formula:

$$w_{ft}^{k} - w_{f,t-s_f}^{k} \le 1 - x_{f'f}$$

At the same time, the decision variable $x_{f'f}$ produces a new constraint formula, indicating that only one flight can connect to the previous flight, the formula is as follows:

$$\sum_{f\in R_{f'}} x_{f'f} = 1$$

Finally, considering the actual bank, consider the flight as a bank, add new 0-1 decision variables $y_{B,t}$ and $z_{B,t}$, and use them to constrain the earliest arriving flight and the latest arriving flight in the bank. Therefore, new constraints are generated as follows:

$$y_{B,t} - w_{ft}^{k} \geq 0 \ \ ; \ \ z_{B,t} - w_{ft}^{k} \geq 0$$

$$y_{B,t} - y_{B,t\text{-}1} \geq 0 \ \ ; \ \ z_{B,t} - z_{B,t\text{-}1} \geq 0$$

Due to the constraints on the bank, the change in the expression of the objective function is caused. The original objective function is simplified to the following form:

$$\min\left[\sum_{t\in T} t(z_{B,t} - z_{B,t-1}) - \sum_{t\in T} t(y_{B,t} - y_{B,t-1})\right]$$

The **extended TFMP model** is organized as follows:

$$\min\left[\sum_{t\in T} t(z_{B,t} - z_{B,t-1}) - \sum_{t\in T} t(y_{B,t} - y_{B,t-1})\right]$$

Subject to:

$$\alpha_{kt}^{i} \sum_{f:t\in T_f^k, k=P(f,1)} (w_{ft}^{k} - w_{f,t-1}^{k}) + \beta_{kt}^{i} \sum_{f:t\in T_f^k, k=P(f,N_f)} (w_{ft}^{k} - w_{f,t-1}^{k}) \leq \gamma_{kt}^{i} \tag{1}$$

$$\sum_{f:P(f,i)=j, P(f,i+1)=j', i<N_f} (w_{ft}^{j} - w_{f,t}^{j'}) \leq S_j(t) \tag{2}$$

$$w_{f,t+l_{fj}}^{j'} - w_{f,t}^{j} \leq 0 \tag{3}$$

$$w_{ft}^{k} - w_{f,t-s_f}^{k} \leq 1 - x_{f'f} \tag{4}$$

$$\sum_{f\in R_{f'}} x_{f'f} = 1 \tag{5}$$

$$w_{f,t}^{j} - w_{f,t\text{-}1}^{j} \geq 0 \tag{6}$$

$$y_{B,t} - w_{ft}^{k} \geq 0 \ \ ; \ \ z_{B,t} - w_{ft}^{k} \geq 0 \tag{7}$$

$$y_{B,t} - y_{B,t\text{-}1} \geq 0 \ \ ; \ \ z_{B,t} - z_{B,t\text{-}1} \geq 0 \tag{8}$$

Decision variables are:

w_{ft}^{k} ; $x_{f'f}$; $y_{B,t}$; $z_{B,t}$

2. Case data

Create a virtual aviation system case as shown in Figure 1 below. The case includes three airports A, B, and C, and four sectors of 1, 2, 3, and 4.

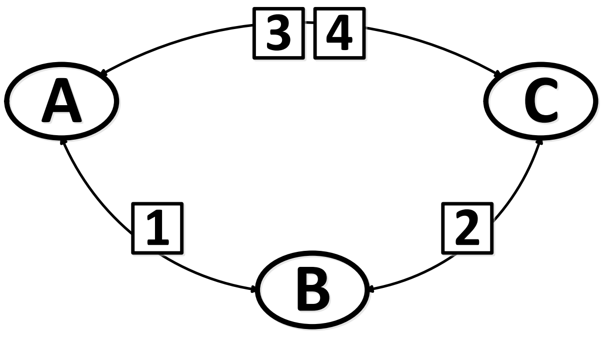


Figure 1 virtual aviation system

Assume that there are 10 flights in the system that need to be deployed. The flight information is in Table 1, including flight number and flight route information. The case does not give the different allocation of the runway, only the ability to accept and send the aircraft, so the service capacity of the airport still use the formula related to $D_k(t)$ and $A_k(t)$, their information is in Table 2. The service capabilities of each sector are in Table 3. The preparing time required for the flight connection is shown in Table 4. The required flight time for the aircraft to fly through each sector is shown in Table 5.

Table 1 Flights route

flight	k=1	k=2	k=3	k=4
C01	A	1	B	
C02	A	3	4	C
C03	C	2	B	
C04	C	3	4	A
C05	B	2	C	
C06	B	1	A	
C07	C	3	4	A

C08	A	1	B	
C09	B	2	C	
C10	A	3	4	C

Table 2 Service capacity in airports

airport	D	A
11	2	1
22	1	1
33	1	1

Table 3 Service capacity in sectors

sector	S
1	2
2	2
3	2
4	3

Table 4 Connect flights' preparing time

flight	s
1	1
2	1
3	1
4	2
5	2
6	2
7	1
8	1
9	1
10	2

Table 5 Flying time through sectors

lfj	1	2	3	4
C01	1	1	1	1
C02	1	1	2	1
C03	1	1	2	1
C04	1	1	2	2

C05	1	1	2	2
C06	1	2	2	2
C07	2	1	2	2
C08	1	1	2	1
C09	1	1	1	1
C10	2	2	2	2

Use the data in the case for programming demonstrations. Try to consider massive data to solve when programming, instead of just a simple set of flights in the case. The model, in this case, can be expressed as:

$$\min\left[\sum_{t\in T} t(z_{B,t}-z_{B,t-1})-\sum_{t\in T} t(y_{B,t}-y_{B,t-1})\right]$$

Subject to:

$$\left\{\begin{array}{l}
(w^1_{1,t}-w^1_{1,t-1})+(w^1_{2,t}-w^1_{2,t-1})+(w^1_{8,t}-w^1_{8,t-1})+(w^1_{10,t}-w^1_{10,t-1})\le D_A \quad \cdots \\
(w^1_{5,t}-w^1_{5,t-1})+(w^1_{6,t}-w^1_{6,t-1})+(w^1_{9,t}-w^1_{9,t-1})\le D_B \quad \cdots \\
(w^1_{3,t}-w^1_{3,t-1})+(w^1_{4,t}-w^1_{4,t-1})+(w^1_{7,t}-w^1_{7,t-1})\le D_C \quad \cdots
\end{array}\right.$$

$$\left\{\begin{array}{l}
(w^4_{4,t}-w^4_{4,t-1})+(w^3_{6,t}-w^3_{6,t-1})+(w^4_{7,t}-w^4_{7,t-1})\le A_A \quad \cdots \\
(w^3_{1,t}-w^3_{1,t-1})+(w^3_{3,t}-w^3_{3,t-1})+(w^3_{8,t}-w^3_{8,t-1})\le A_B \quad \cdots \\
(w^4_{2,t}-w^4_{2,t-1})+(w^3_{5,t}-w^3_{5,t-1})+(w^3_{9,t}-w^3_{9,t-1})+(w^4_{10,t}-w^4_{10,t-1})\le A_C \quad \cdots
\end{array}\right.$$

$$\left\{\begin{array}{l}
(w^2_{1,t}-w^3_{1,t})+(w^2_{6,t}-w^3_{6,t})+(w^2_{8,t}-w^3_{8,t})\le S_1 \quad \cdots \\
(w^2_{3,t}-w^3_{3,t})+(w^2_{5,t}-w^3_{5,t})+(w^2_{9,t}-w^3_{9,t})\le S_2 \quad \cdots \\
(w^2_{2,t}-w^3_{2,t})+(w^3_{4,t}-w^4_{4,t})+(w^3_{7,t}-w^4_{7,t})+(w^2_{10,t}-w^3_{10,t})\le S_3 \quad \cdots \\
(w^3_{2,t}-w^4_{2,t})+(w^2_{4,t}-w^3_{4,t})+(w^2_{7,t}-w^3_{7,t})+(w^3_{10,t}-w^4_{10,t})\le S_4 \quad \cdots
\end{array}\right.$$

Programming Ideas And Steps

An Xu

$$\begin{cases} w_{1,t+1}^{3} - w_{1,t}^{2} \le 0 \\ w_{2,t+2}^{3} - w_{2,t}^{2} \le 0;\quad w_{2,t+1}^{4} - w_{2,t}^{3} \le 0 \\ w_{3,t+1}^{3} - w_{3,t}^{2} \le 0 \quad \cdots \end{cases}$$

$$\begin{cases} w_{5,t}^{1} - w_{1,t\text{-}1}^{3} \le 1\text{-}x_{15};\quad w_{6,t}^{1} - w_{1,t\text{-}1}^{3} \le 1\text{-}x_{16};\quad w_{9,t}^{1} - w_{1,t\text{-}1}^{3} \le 1\text{-}x_{19} \\ w_{3,t}^{1} - w_{2,t\text{-}1}^{4} \le 1\text{-}x_{23};\quad w_{4,t}^{1} - w_{2,t\text{-}1}^{4} \le 1\text{-}x_{24};\quad w_{7,t}^{1} - w_{2,t\text{-}1}^{4} \le 1\text{-}x_{27} \\ \cdots \end{cases}$$

$$\begin{cases} x_{15} + x_{16} + x_{19} = 1 \\ x_{23} + x_{24} + x_{27} = 1 \quad \cdots \end{cases}$$

$$\begin{cases} w_{1,t}^{2} - w_{1,t\text{-}1}^{2} \le 0 \\ w_{2,t}^{2} - w_{2,t\text{-}1}^{2} \le 0;\quad w_{2,t}^{3} - w_{2,t\text{-}1}^{3} \le 0 \\ \cdots \end{cases}$$

$$\begin{cases} y_t - w_{1,t}^{3} \ge 0;\quad y_t - w_{2,t}^{4} \ge 0;\quad y_t - w_{3,t}^{3} \ge 0 \quad \cdots \\ z_t - w_{1,t}^{3} \le 0;\quad z_t - w_{2,t}^{4} \le 0;\quad z_t - w_{3,t}^{3} \le 0 \quad \cdots \\ y_t - y_{t\text{-}1} \ge 0 \;;\; z_t - z_{t\text{-}1} \ge 0 \end{cases}$$

Note: The programming process is not written in accordance with the above formulas, these formulas are only used for understanding and display.

3. Programming thought

The programming software uses IBM CPLEX, and the programming language selects the OPL language. The focus of this programming is on the import of multi-dimensional case data, data processing, and the input of parameters and conditions in the model. The difficulty of programming is the processing of multidimensional data and the use of retrieval, the rules of the OPL language, and the correct input of the model. Because the model contains a large number of constraints, it requires correct and effective logic to solve the problem.

The program is mainly divided into the call of parameters, the processing of data, the preparation of the objective function and the input of constraints. (1) The object called by the parameter is all the known parameter information in the case. The purpose of the call is to organize the large-scale data into a form that the language program can recognize, in preparation for the operation. (2) The input parameters need to be processed because the different rules of the information need to be processed. For the convenience of post-calculation, I pre-processed all the data that I couldn't directly input but needed for the calculation. The focus of this section is on the need to ensure that the logic and the call format are correct. (3) The preparation of the objective function only needs to be input according to the correct function in the model, pay attention to the setting of the loop. (4) Constraints need to be strictly followed by the correct logic when inputting, and programming needs to address the complexity of large-scale constraints. Requires strict correspondence when calling parameters, and a clear loop method. Only by ensuring the correctness of the program ideas and logic can we obtain reliable calculation results.

The initial parameters called are all the parameters in Table 1 to Table 5. The remaining data to be processed includes the k value corresponding to the flight endpoint, which is the Nf value in the article, the set of sectors, and the set of flights that can be connected. The processed data set should be identical to Tables 6 and 7 below.

Table 6 Set of Nf value and sectors

flight	Nf	J
C01	3	2
C02	4	2, 3
C03	3	2
C04	4	2, 3
C05	3	2

C06	3	2
C07	4	2, 3
C08	3	2
C09	3	2
C10	4	2, 3

Table 7 Set of connected flights

flight	Rf			
C01	C05	C06	C09	
C02	C03	C04	C07	
C03	C05	C06	C09	
C04	C01	C02	C08	C10
C05	C03	C04	C07	
C06	C01	C02	C08	C10
C07	C01	C02	C08	C10
C08	C05	C06	C09	
C09	C03	C04	C07	
C10	C03	C04	C07	

In addition to this, it is necessary to classify the departure and arrival airports of each flight. The processed flights' departure from the different airport is shown in Table 8. The flights' arrival airport collection is shown in Table 9. The sectors also need to be organized, which should be three-dimensional, and the processed data are shown in Table 10.

Table 8 Flights departure from the different airport

airport	start(f)			
A	C01	C02	C08	C10
B	C05	C06	C09	
C	C03	C04	C07	

Table 9 Flights arrive at a different airport

airport	end(f)			
A	C04	C06	C07	
B	C01	C03	C08	
C	C02	C05	C09	C10

Table 10 Flights arrive at different sector

k	sectors	K		
2	j=1	C01	C06	C08
	j=2	C03	C05	C09
	j=3	C02	C10	
	j=4	C04	C07	
3	j=1			
	j=2			
	j=3	C04	C07	
	j=4	C02	C10	

With the logic of the above case, the model can be summarized and programmed. The following is a detailed description of the programming steps.

4. Program writing steps

- **Parameter input**

The parameters that need to be input in this part include the total number of flights, the route of the flight (P[f][k]), the service capacity of the airport (D[a], A[a]), the service capacity of the sector (S[j]), the preparing time required for the flight connection (s[f]), and the travel time of the aircraft within the sector (l[f][j]). In order to ensure the universal applicability of the written program, in the process of writing, the data are all retrieved and assigned from the original data, and the case data are not used in the process.

The input method of the parameter is: use the OPL language to retrieve the data in Excel. The specific data type definitions of this part of the parameters and the procedure of the retrieval procedure are as follows:

<In .mod document>

```
{int} f =...;
{int} k = ...;
{int} j = ...;
{int} a = ...;

int P[f][k] =...;
int s[f] =...;
int l[f][j] =...;
int D[a] =...;
int A[a] =...;
int S[j] =...;
range t =1..30;
range F=1..10;
```

<In .dat document>

```
SheetConnection sheet("Testcplex.xlsx");
f from SheetRead(sheet,"flight!G2:G11");
k from SheetRead(sheet,"flight!H1:K1");
P from SheetRead(sheet,"flight!H2:K11");
s from SheetRead(sheet,"parameter!G2:G11");
j from SheetRead(sheet,"parameter!B1:E1");
l from SheetRead(sheet,"parameter!B2:E11");
a from SheetRead(sheet,"parameter!I2:I4");
D from SheetRead(sheet,"parameter!J2:J4");
A from SheetRead(sheet,"parameter!K2:K4");
S from SheetRead(sheet,"parameter!J8:J11");
```

All the original data has been entered here.

- **Parameter processing**

It is necessary to consider the set of connected flights R_f in the model, the set of departure N_f, and the set of sectors through which the flight passes J_f. At the same time, the data is sorted into different sets according to the requirements of the model, and the airport departure flight collection $start(a, f)$, the airport arrival flight collection $end(a, f)$, and light set through the sectors $K_{(k,j,f)}$ collection are obtained.

Programming Ideas And Steps

An Xu

First, determine the end position of the flight, ie $k = N_f$ The N_f array is determined by whether the information $P_{f,k}$ is empty. The specific procedures are:

```
int N[f];
execute {
    for(var i in f)
        for(var m in k){

        if(P[i][m]==0)
     N[i]=m-1;
 else N[i]=m;
 }
```

Next, find the sector J , in which the flight needs to pass, and the determination of J is obtained by N_f . J indicates the sector to be followed by the flight except for the beginning position (*k*=1) and the ending position (*k*=*Nf*):

```
int J[f];
execute {
  for(var i in f) {
  J[i] = N[i]-1;
   }
```

Then find out the possible connected set R_f for each flight. The criterion is that the departure airport of the next flight is the same as the arrival airport of the previous flight:

```
int R[f][F];
execute {
  for(var i in f) {
   var m = 1;
  for(var n in f)
         if(P[i][N[i]]==P[n][1])
    { R[i][m]=n; m++;
     }
```

Programming Ideas And Steps

An Xu

The airport departure flight set is classified by judging the values in the $P_{f,k}$ set, and the obtained $start(a, f)$ array is a sequence of different airports.

```
int start[a][F];
int p[a];
execute{
     for(var i in a){
         p[i] = 1;    }
     for(var i in f){
         for(var n in a){
             if(P[i][1]==n){
               start[n][p[n]]=i;
               p[n]++;
               break;
             }
```

The airport arrival flight collection is the same as before, judge the value in the $P_{f,k}$ set . The difference is that $k{=}Nf$ needs to be judged, and the $end(a, f)$ array obtained is still in the sequence of different airports. $end(a, f)$

```
int end[a][F];
execute{
     for(var i in a){
         p[i] = 1; }
     for(var i in f){
         for(var n in a){
         if(P[i][N[i]]==n){
           end[n][p[n]]=i;
           p[n]++;
           break;
         }
```

The flight set through the sector $K_{(k,j,f)}$ is divided into a set by the flight of the sector. The specific operation process is:

```
int K[k][j][F];
int area[j];
```

```
execute{
    for(var i in j){
        area[i] = 1;
    }
    for(var l in k){
        for(var i in f){
            for(var n in j)
            if(P[i][l]==n){
              K[l][n][area[n]]=i;
              area[n]++;
              break;
            }
```

- **Definition of decision variables**

The decision variables in the model include four types: w_{ft}^{k}, $x_{f'f}$, y_t, and z_t , all of them are 0-1 variables. The specific commands are:

```
dvar int+ z[t] in 0..1;
dvar int+ w[f][t][k] in 0..1;
dvar int+ y[t] in 0..1;
dvar int+ x[f][F] in 0..1;
```

- **The input of the objective function**

The objective function is the same as the instruction in the model. Input according to the language input method:

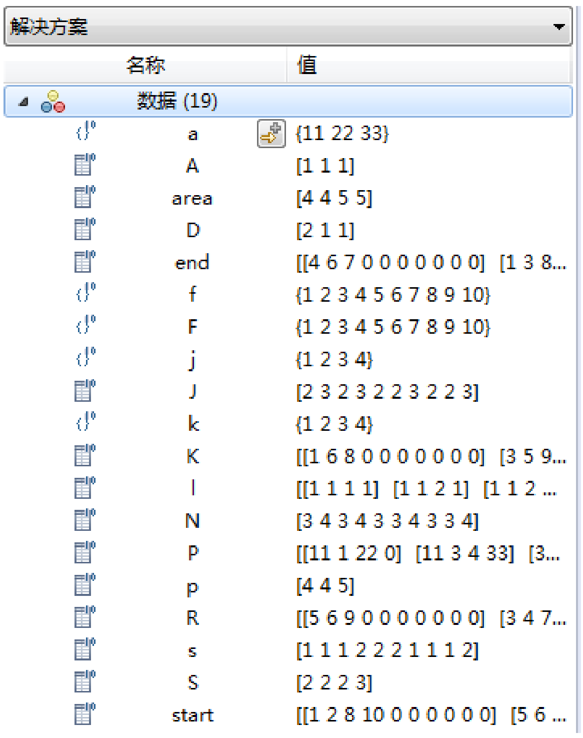
```
minimize
    sum(t in 1..30)(t*(z[t]-z[t-1]))-sum(t in 1..30)(t*(y[t]-y[t-1]));
```

- **Input of Constraints**

Since the parameters that need to be called have been defined one by one, the constraint part only needs to be input according to the language of the program and the parameters in the model. Note that the loop and assignment problems between arrays of parameters. The specific procedures are not repeated here.

- **Program test**

Check the input parameters according to the above program. The output is shown in Figure 2 below and can be seen to be consistent with the expected effect. Proof of the input and processing of the program is practicable.



解决方案

名称	值
数据 (19)	
a	{11 22 33}
A	[1 1 1]
area	[4 4 5 5]
D	[2 1 1]
end	[[4 6 7 0 0 0 0 0 0 0] [1 3 8...
f	{1 2 3 4 5 6 7 8 9 10}
F	{1 2 3 4 5 6 7 8 9 10}
j	{1 2 3 4}
J	[2 3 2 3 2 2 3 2 2 3]
k	{1 2 3 4}
K	[[1 6 8 0 0 0 0 0 0 0] [3 5 9...
l	[[1 1 1 1] [1 1 2 1] [1 1 2 ...
N	[3 4 3 4 3 3 4 3 3 4]
P	[[11 1 22 0] [11 3 4 33] [3...
p	[4 4 5]
R	[[5 6 9 0 0 0 0 0 0 0] [3 4 7...
s	[1 1 1 2 2 2 1 1 1 2]
S	[2 2 2 3]
start	[[1 2 8 10 0 0 0 0 0 0] [5 6 ...

Figure 2 Input data test

<u>TFMRP</u>

1. Understanding of formulas

The TFMRP model extends the multipath decision on the original model, it is no longer that flights travel on a defined single path.

There are two ways to expand the path. The first is to consider more options for the route according to the origin and destination of the flight. The alternative path is denoted by *r*. When making a path selection, it is necessary to separately calculate the target value of each path plan, and finally select the driving path according to the output decision variable. At this time, the decision variable w_{ft}^{k} is no longer a three-dimensional variable, it is necessary to increase the path decision to become a four-dimensional parameter w_{ft}^{jr} . Decision variables lead to an increase in constraints to ensure that only one complete route is selected. The added constraints are:

$$\sum_r w_{ft}^{jr} = w_{ft}^{j}$$

The second extension method is to select according to the connection relationship between the sectors, and it is necessary to introduce the sets N_{fj} and P_{fj} respectively to represent the next sector that can be connected and the set of the previous sector. Use these two sets to represent all possible route plans. At this point the decision variable becomes $w_{ft}^{jj'}$, increasing the constraints of the path selection:

$$\sum_{j'} w_{ft}^{jj'} = w_{ft}^{j}$$

2. Case data

According to the 10 flights, in this case, the flight information in Table 1 is no longer valid, and it is necessary to determine the feasible route for each flight. For example, for flight C01, an alternative can be given as shown in Table 11 below.

Table 10 Alternative routes of flight C01

C01	1	2	3	4	5
r1	A	1	B		
r2	A	3	2	B	
r3	A	3	1	B	
r4	A	3	4	2	B
r5	A	3	4	1	B

Or according to the sector alternative method in the second method, the added sets N_{fj} and P_{fj} for flight C01 should be:

$$N(1,1)=\{B\};\quad N(1,2)=\{B\};\quad N(1,3)=\{1,2,4\};\quad N(1,4)=\{2\}$$

$$P(1,1)=\{A\};\quad P(1,2)=\{3,4\};\quad P(1,3)=\{A\};\quad P(1,4)=\{3\}$$

This expression can also determine the set of alternative paths for flight C01. In this way, you can find alternative flight routes for each flight. When determining the decision variables, the determined driving route can be obtained, which makes the model more flexible and can adapt to the flight allocation management under more circumstances.

3. Program writing thought

The main body of the program is still the same as the TFMP model. The difference is that the decision variables need to be redefined, especially for w_{ft}^{k}, which should become a 4-dimensional variable. In the constraint, it needs to rewrite only when the route involving the j sector is involved. The difficulty is that the determination of the set of candidate sectors, as well as the variable increment dimension, may cause some logical problems.