CS 5727 Optimization Method, Project

Roger Wang, rw575

December 3, 2018

1 Executive Summary

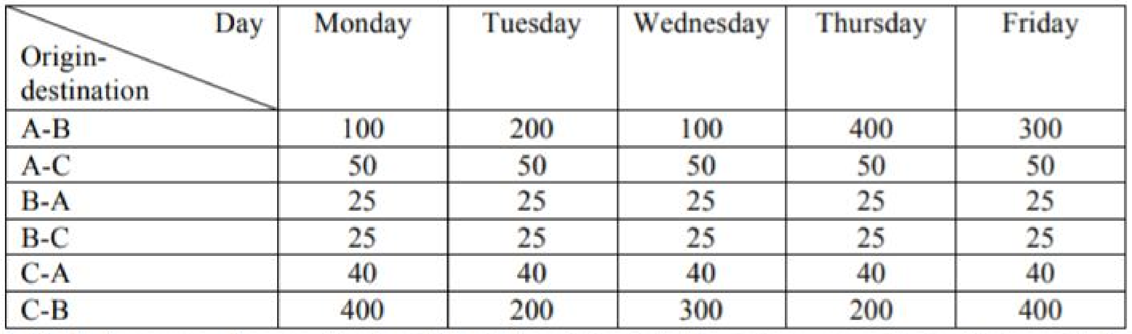
This project is about developing an optimization model that finds a weekly aircraft movement cycle with minimum total cost for Express Air, while ensuring that all cargo is delivered from its origin to its destination.

2 Problem Overview

Express Air operates an aircraft fleet to run a cargo business among three airports, airports A, B and C. Each day, a certain amount of cargo that needs to be delivered between each origin-destination airport arrives into the system. There is a consistent weekly pattern for the cargo that needs to be delivered. Express Air can deliver a cargo or move the empty aircraft between airports. The travel time is assumed to be one full day and there is an extra cost for holding the cargo to the next day and moving the empty aircraft. Express Air needs a weekly schedule for the aircraft to minimize the extra cost while delivering all the cargo. In particular, the number of aircraft that moves into an airport by Friday evening should be what is available at the same airport Monday morning so that the movement cycle for the aircraft can be repeated every week.

3 Data Description

There is a consistent weekly pattern for the cargo that needs to be delivered. Table 1 shows the amount of cargo arriving into the system on each day of the week that needs to be delivered between each origin-destination airport. All of the quantities in the table are in full aircraft loads. There are 1200 aircraft in total. Each aircraft can carry one aircraft load – no more, no less. If there is not enough aircraft to deliver the cargo, the cargo can be held at an extra cost of 10 per day. Express Air can move the empty aircraft between airports for future delivery with an extra cost. The extra cost for moving between A and B is 7. The extra cost for moving between A and C is 3. The extra cost moving between B and C is 3.



Origin-destination \ Day	Monday	Tuesday	Wednesday	Thursday	Friday
A-B	100	200	100	400	300
A-C	50	50	50	50	50
B-A	25	25	25	25	25
B-C	25	25	25	25	25
C-A	40	40	40	40	40
C-B	400	200	300	200	400

Table 1: Amounts of cargo (in aircraft loads) arriving into the system on each day that need to be carried between each origin-destination airport

4 Optimization Model

4.1 math symbol:

N: set of all the airports
N = {A, B, C}
T: set of all the weekdays
T = {1 (Monday), 2 (Tuesday), 3 (Wednesday), 4 (Thursday), 5(Friday)}
c_{ij}: cost of carrying a load from i to j. $(i, j \in N)$
$c_{AB} = c_{BA} = 7$
$c_{AC} = c_{CA} = 3$
$c_{BC} = c_{CB} = 6$
$c_{AA} = c_{BB} = c_{CC} = 0$
s_{ijt}: amount of new cargo arrived that need to be carried from i to j on day t. $(i, j \in N \quad t \in T)$
The value is given by *table 1.*
x_{ijt}: number of aircraft that carry load from i to j on day t. $(i, j \in N \quad t \in T)$
y_{ijt}: number of aircraft without load that fly from i to j on day t. $(i, j \in N \quad t \in T)$
d_{ijt}: total amount of cargo that need to be carried from i to j on day t. $(i, j \in N \quad t \in T)$

4.2 linear program:

$$\text{minimize } 10 \cdot \sum_{i\in N}\sum_{j\in N}\sum_{t\in T}(d_{ijt} - x_{ijt}) + \sum_{i\in N}\sum_{j\in N}\sum_{t\in T}(c_{ij} \cdot y_{ijt}) + 20 \cdot \sum_{i\in N}\sum_{j\in N}(d_{ij5} - x_{ij5})$$

s.t.

$$\sum_{i\in N}\sum_{j\in N} x_{ij1} + \sum_{i\in N}\sum_{j\in N} y_{ij1} = 1200 \tag{1}$$

$$\sum_{j\in N} x_{ijt} + \sum_{j\in N} y_{ijt} = \sum_{j\in N} x_{ji(t-1)} + \sum_{j\in N} y_{ji(t-1)} \qquad \forall i \in N \quad t \in \{2, 3, 4, 5\} \tag{2}$$

$$\sum_{j\in N} x_{ij1} + \sum_{j\in N} y_{ij1} = \sum_{j\in N} x_{ji5} + \sum_{j\in N} y_{ji5} \qquad \forall i \in N \tag{3}$$

$$x_{ijt} \leqslant d_{ijt} \qquad \forall i, j \in N \quad t \in T \tag{4}$$

$$d_{ijt} = d_{ij(t-1)} - x_{ij(t-1)} + s_{ijt} \qquad \forall i,j \in N \quad t \in \{2,3,4,5\} \tag{5}$$

$$d_{ij1} = d_{ij5} - x_{ij5} + s_{ij1} \qquad \forall i,j \in N \tag{6}$$

$$d_{ijt} \geqslant 0,\ x_{ijt} \geqslant 0,\ y_{ijt} \geqslant 0 \qquad \forall i,j \in N \quad t \in T \tag{7}$$

The first part of the objective function is the extra cost for holding the cargo during weekday.
The second part of the objective function is the extra cost for moving the empty aircraft.
The third part of the objective function is the extra cost for holding the cargo on Saturday and Sunday if some amount of cargo on Friday needs to be transitioned to the next week.

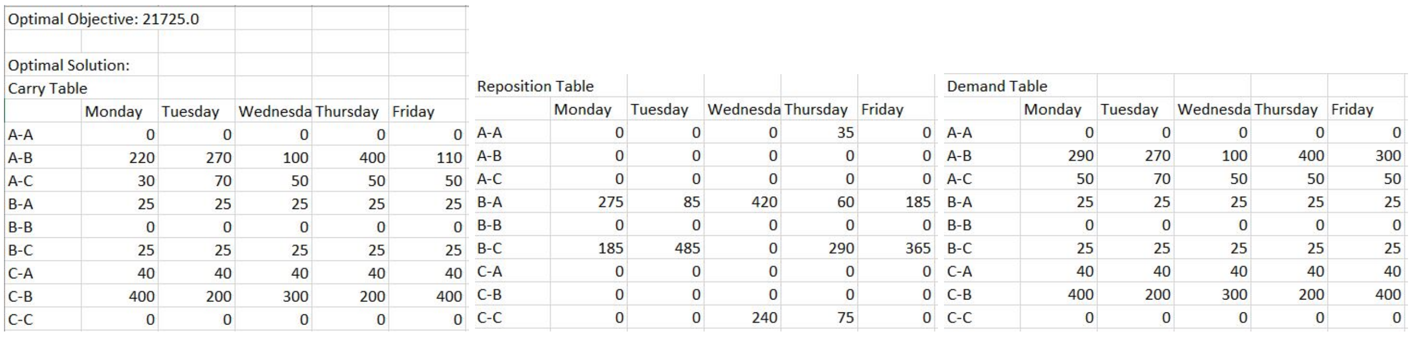
Constraint 1 specifies that the total number of aircraft is 1200.
Constraints 2 & 3 specify that during each weekday, the amount of aircraft leaving at each airport equals the amount of aircraft arrived at at that airport in the previous day. Constraint 3 is added on Monday to form a cycle.
Constraint 4 specifies that the aircraft can't carry more cargo than the demanded amount of cargo to be carried during each weekday, at each route between airports.
Constraint 5 & 6 specify that during each weekday, at each route between airports, the demanded amount of cargo to be carried equals previous day's unfinished demand plus the new demand on that day.

4.3 python3 code:

```python
from gurobipy import *

# x is a 3D matrix of decision variables.
# x[i][j][t] means number of aircraft that carry load from i to j on day t.
x = []

# y is a 3D matrix of decision variables.
# y[i][j][t] means number of aircraft without load that fly from i to j on day t.
y = []

# d is a 3D matrix of decision variables.
# d[i][j][t] means total amount of cargo that need to be carried from i to j on day t.
d = []

# cost is a 2D matrix of cost values.
# cost[i][j] means the cost of carrying a load from i to j.
cost = [[0, 7, 3],
        [7, 0, 6],
        [3, 6, 0]]

# supply is a 3D matrix of newly arrived cargo values.
# supply[t][i][j] means the amount of new cargo arrived that need to be carried from i to j on day
supply = [[[0, 100, 50], [25, 0, 25], [40, 400, 0]],
          [[0, 200, 50], [25, 0, 25], [40, 200, 0]],
          [[0, 100, 50], [25, 0, 25], [40, 300, 0]],
          [[0, 400, 50], [25, 0, 25], [40, 200, 0]],
          [[0, 300, 50], [25, 0, 25], [40, 400, 0]]]

airport_num = 3 # airport A, B, C
```

```python
weekday_num = 5 # Mon, Tue, Wed, Thu, Fri
aircraft_num = 1200 # total number of aircraft

model = Model("project")

# add decision variables for x
for i in range(airport_num):
    mx = []
    for j in range(airport_num):
        li = []
        for t in range(weekday_num):
            li.append(model.addVar(vtype=GRB.INTEGER, name='x'+str(i)+str(j)+str(t)))
        mx.append(li)
    x.append(mx)

# add decision variables for y
for i in range(airport_num):
    mx = []
    for j in range(airport_num):
        li = []
        for t in range(weekday_num):
            li.append(model.addVar(vtype=GRB.INTEGER, name='y'+str(i)+str(j)+str(t)))
        mx.append(li)
    y.append(mx)

# add decision variables for d
for i in range(airport_num):
    mx = []
    for j in range(airport_num):
        li = []
        for t in range(weekday_num):
            li.append(model.addVar(vtype=GRB.INTEGER, name='d'+str(i)+str(j)+str(t)))
        mx.append(li)
    d.append(mx)

model.update()

# create objective function that minimizes the total cost
objExpr = LinExpr()
# cost of holding the cargo in the weekday
for i in range(airport_num):
    for j in range(airport_num):
        for t in range(weekday_num):
            objExpr += 10 * (d[i][j][t] - x[i][j][t])
# cost of repositioning the aircraft
for i in range(airport_num):
    for j in range(airport_num):
        for t in range(weekday_num):
            objExpr += (cost[i][j] * y[i][j][t])
# extra cost of holding the cargo to the next week
for i in range(airport_num):
    for j in range(airport_num):
        objExpr += 20 * (d[i][j][4] - x[i][j][4])

model.setObjective(objExpr, GRB.MINIMIZE)

# create constraint for the total number of aircraft
constExpr = LinExpr()
for i in range(airport_num):
    for j in range(airport_num):
        constExpr += x[i][j][1] + y[i][j][1]
model.addConstr(constExpr, GRB.EQUAL, aircraft_num, 'total aircraft')

# create constraints that aircraft out on day t == aircraft in on day t-1, t>=1
for i in range(airport_num):
    for t in range(1, weekday_num):
        constExpr = LinExpr()
        for j in range(airport_num):
            constExpr += x[i][j][t] + y[i][j][t] - x[j][i][t-1] - y[j][i][t-1]
        model.addConstr(constExpr, GRB.EQUAL, 0, 'balance airport'+str(i)+' day '+str(t))

# create constraints that aircraft out on day 0 == aircraft in on day 4
for i in range(airport_num):
    constExpr = LinExpr()
    for j in range(airport_num):
        constExpr += x[i][j][0] + y[i][j][0] - x[j][i][4] - y[j][i][4]
    model.addConstr(constExpr, GRB.EQUAL, 0, 'balance airport'+str(i)+' day0')

# create constraints that aircraft carry cargo out <= cargo demand
for i in range(airport_num):
    for j in range(airport_num):
        for t in range(weekday_num):
            constExpr = LinExpr()
            constExpr = x[i][j][t] - d[i][j][t]
            model.addConstr(constExpr, GRB.LESS_EQUAL, 0, 'carry constraint '+str(i)+' to '+str(j)

# create constraints that cargo demand on day t = cargo demand on day t-1 + supply - carried cargo
for i in range(airport_num):
    for j in range(airport_num):
        for t in range(1, weekday_num):
            constExpr = LinExpr()
            constExpr = d[i][j][t] - d[i][j][t-1] + x[i][j][t-1] - supply[t][i][j]
            model.addConstr(constExpr, GRB.EQUAL, 0, 'demand value '+str(i)+' to '+str(j)+' on day

# create constraints that cargo demand on day 0 = cargo demand on day 4 + supply - carried cargo
for i in range(airport_num):
    for j in range(airport_num):
        constExpr = LinExpr()
        constExpr = d[i][j][0] - d[i][j][4] + x[i][j][4] - supply[0][i][j]
        model.addConstr(constExpr, GRB.EQUAL, 0, 'demand value '+str(i)+' to '+str(j)+' on day0')

model.update()

# check whether linear program is constructed correctly
model.write('project.lp')

model.optimize()

# output optimal objective and decision variables' value to a file
f = open('solution.csv', 'w')
f.write('Optimal Objective: '+str(model.objVal)+'\n')
f.write('\nOptimal Solution:\n')
f.write('Carry Table\n')
f.write(',Monday,Tuesday,Wednesday,Thursday,Friday\n')
airport_code = {0: 'A', 1: 'B', 2: 'C'}
for i in range(airport_num):
    for j in range(airport_num):
        f.write(airport_code[i]+'-'+airport_code[j])
        for t in range(weekday_num):
            f.write(','+str(int(x[i][j][t].x)))
        f.write('\n')
f.write('\nReposition_Table\n')
f.write(',Monday,Tuesday,Wednesday,Thursday,Friday\n')
for i in range(airport_num):
    for j in range(airport_num):
        f.write(airport_code[i]+'-'+airport_code[j])
        for t in range(weekday_num):
            f.write(','+str(int(y[i][j][t].x)))
        f.write('\n')
f.write('\nDemand_Table\n')
f.write(',Monday,Tuesday,Wednesday,Thursday,Friday\n')
for i in range(airport_num):
    for j in range(airport_num):
        f.write(airport_code[i]+'-'+airport_code[j])
        for t in range(weekday_num):
            f.write(','+str(int(d[i][j][t].x)))
        f.write('\n')
f.close()
```

4.4 output:



Optimal Objective: 21725.0

Optimal Solution:

Carry Table

	Monday	Tuesday	Wednesda	Thursday	Friday
A-A	0	0	0	0	0
A-B	220	270	100	400	110
A-C	30	70	50	50	50
B-A	25	25	25	25	25
B-B	0	0	0	0	0
B-C	25	25	25	25	25
C-A	40	40	40	40	40
C-B	400	200	300	200	400
C-C	0	0	0	0	0

Reposition Table

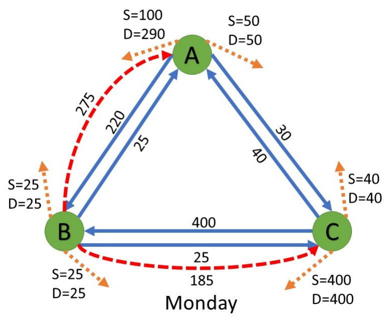
	Monday	Tuesday	Wednesda	Thursday	Friday
A-A	0	0	0	35	0
A-B	0	0	0	0	0
A-C	0	0	0	0	0
B-A	275	85	420	60	185
B-B	0	0	0	0	0
B-C	185	485	0	290	365
C-A	0	0	0	0	0
C-B	0	0	0	0	0
C-C	0	0	240	75	0

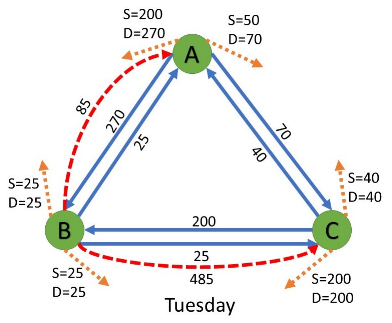
Demand Table

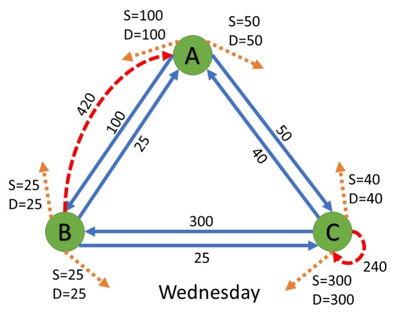
	Monday	Tuesday	Wednesda	Thursday	Friday
A-A	0	0	0	0	0
A-B	290	270	100	400	300
A-C	50	70	50	50	50
B-A	25	25	25	25	25
B-B	0	0	0	0	0
B-C	25	25	25	25	25
C-A	40	40	40	40	40
C-B	400	200	300	200	400
C-C	0	0	0	0	0

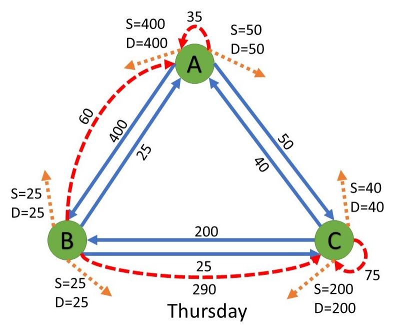
5 Analysis:

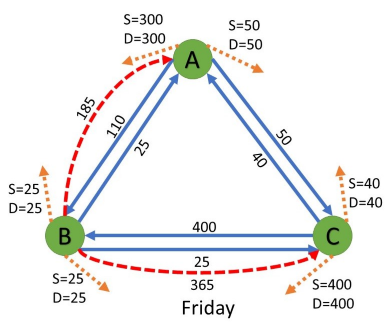
Based on the output tables, I draw five diagrams representing each day's schedule. The blue arrow is the amount of cargo carried by aircraft. The red dashed arrow is the empty aircraft movement. The orange dotted arrow represents two values to the pointed airport. "S" is the newly arrived amount of cargo. "D" is the total amount of cargo to be carried.





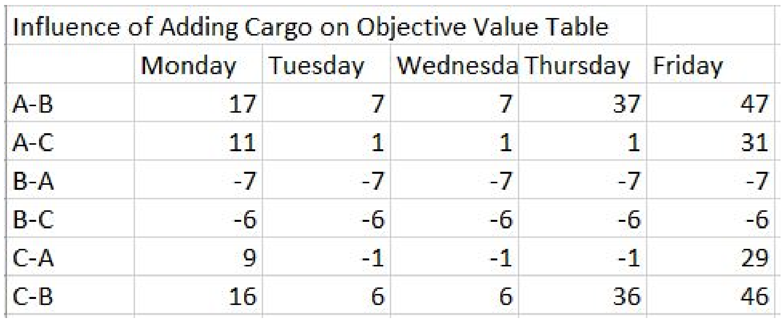






From the diagrams, we can see that carrying cargo on route B-C, B-A, C-A can be done on a daily basis. One major part of the cost comes from airport B frequently sending empty aircraft back to A and C. Another major part of the cost comes from not all the cargo can be delivered on Friday. $D_{1AB} - S_{1AB} = 290 - 100 = 190$ cargoes on route A-B needs to be held to the next week to be delivered. $220 - S_{1AB} = 120$ of them will be delivered on Monday. $270 - S_{2AB} = 270 - 200 = 70$ of them will be delivered on Tuesday. Those 190 cargo's holding cost will be $3 \times 10 \times 120 + 4 \times 10 \times 70 = 6400$.

The influence of adding cargo and adding aircraft on cost is shown below.



Influence of Adding Cargo on Objective Value Table					
	Monday	Tuesday	Wednesda	Thursday	Friday
A-B	17	7	7	37	47
A-C	11	1	1	1	31
B-A	-7	-7	-7	-7	-7
B-C	-6	-6	-6	-6	-6
C-A	9	-1	-1	-1	29
C-B	16	6	6	36	46

If the number of aircraft increase by 1, the extra cost will decrease by 40.

Adding cargo on Friday route A-B, A-C, C-A, C-B incurs the most cost compared with the same route on other weekdays. Adding cargo on Thursday route A-B and C-B, Friday route A-B, A-C, C-A and C-B incurs significant cost.

6 Further Application

The method of solving this problem can be applied to solve many similar problems which uses resource to generate value over certain period of time: Uber driver assignment; public transit; restaurant service; shopping mall supply etc.