ISSN: 0974-5823 Vol. 7 No. 5 May, 2022

International Journal of Mechanical Engineering

Applications of Operational Research in Airline Management and Scheduling

Vedika Dixit [1]

[1] Prof. LN Das, Delhi Technological University, Delhi, India.

Abstract - This study base on AIRINDIA Airline Management and Scheduling examined certain well-known problems and their solutions, as well as current research in the field and developing areas of future significance. The paper has divided into four sections. In the first, the Vogel's approximation and the Modi approach is used to create additional routes. In the second, crew is assigned to different flights in order to optimise profit. In the third case, aircraft routing problem is solved, and in the fourth, an aircraft maintenance problem in which 24 activities opted and determine which are critical or not. It addresses airline scheduling flaws, aims to boost crew member productivity through a crew assignment model, and reduces time spent on non-critical operations, resulting in a rise in airline profit.

Keywords : *Airline Management, Airline Scheduling, Vogel's Approximation Method, Dijkstra's Algorithm, Hungarian Assignment Model, Critical Path Analysis, Aircraft Maintenance Problem.*

INTRODUCTION

Operations research has aided the infrastructure and the aviation industry in maintaining strong percentages of growth and moving from a one-of-a-kind item catering to a select customer to a mass-market service business. It aids the industry in evolving in order to must be able to compete successfully in the marketplace and suit the diverse needs of the consumers. The Indian airline industry specifically AIR INDIA is going through a difficult period. Airlines operating margins have been slashed. Many operating expenses, including as fuel, operating lease payments, repair/maintenance, and a huge number of aircraft and crew members, remain unutilized, meaning that they are prone to loss of epic proportions for AIR INDIA and are frequently beyond their control.

This paper is based on the spectacular divestiture of Air India, which ultimately failed. The lack of a management system is AIRINDIA's main concern. They fly more on routes where customer's demand is low and less on routes where demand is high. They have a significant number of aircrafts and crew on reserve, all of which are either unused or used without prior planning. The paper attempts to solve the problems that the Air India Airline faces in a number of situations by developing four separate cases. To tackle their problem, Vogel's Approximation approach and Modi method, Hungarian Method, Dijkstra's Algorithm, and Critical Path Analysis are primarily used.

PROBLEM STATEMENT

Schedule planning involves designing future aircraft and crew timetables in order to enhance airline profitability. A network of flights, various aircraft types, gate, airport slot, and air traffic control restrictions, noise curfews, maintenance requirements, crew work rules, and competitive, dynamic environments in which passenger demand is uncertain and pricing strategies are complex, all add to the complexity of this problem. Unsurprisingly, no optimization model has been proposed or solved to properly handle this complex design challenge. Because of its unmanageable size and complexity, the problem has been broken down into a series of subproblems, which are generally defined as follows:

Route Selection:

It entails determining which aircrafts will serve specific routes, as well as whether or not that route should be included in the airlines' network.

Crew scheduling:

Itis the process of deciding which crews to allocate to each flight in order to reduce crew costs.

Routing:

It is the process of determining which path is best for the aircrafts in order to maximise profit.

Aircraft Maintain Problem:

It is the process of determining the aircrafts to guarantee that maintenance requirements are met.

By addressing the sub problems in succession, and confining the solutions to following problems based on the solutions to previous problems, suboptimal but workable aircraft and crew plans are created. These sub problems are nevertheless large-scale and complex, despite being lesser and less complicated than the total problem. Indeed, OR theorists and practitioners have been building models and algorithms to tackle them for decades, with substantial success and effect.

The solution approach is discussed in the next section, which includes four different airline cases.

Solution Approach

3.1 Case 1: Route Selection

Airlines must conduct extensive market research and demand studies in order to determine whether they can profitably operate the route while also recognising operational restrictions.

In this case, its aims to enhance the process of deciding which destinations to include by using a Transportation Problem (TP) Model, which is a type of Linear Programming Problem (LPP) [1,2]. The TP Model aims to optimise available routes from a specific origin to a specific destination in order to obtain the lowest possible cost.

To demonstrate this, the paper uses the real-situation of an AIR INDIA airline that is aiming to expand its route network. Many airlines use a hub-and-spoke model to provide links to smaller cities via focus cities on international routes. As a result, Table (1) shows data on profit earned per seat sold on a specific sector, as well as demand for destinations across the country, while taking capacity constraints at already overburdened hubs into account.

Table 1: Different data of profit earned between different countries.

	Hongkong	Sydney	Amsterdam	Moscow	Toronto	Bangkok	Lisbon	Supply
Delhi	116	31	71	19	49	47	48	1200
Mumbai	35	12	108	70	49	89	90	900
Chandigarh	54	14	65	19	105	59	68	900
Hyderabad	74	109	58	48	39	51	20	800
Ahmedabad	100	102	90	30	20	58	68	700
Demand	550	930	1250	620	900	800	700	

To make the above-mentioned information into a TP issue, a dummy row is introduced to balance supply and demand, allowing the answer to be solved. Furthermore, profits are converted to 'Opportunity Loss' values, which are defined as the amount missed when compared to taking the most profitable route. The Vogel's Approximation Method (VAM)

[1, 2] is an iterative approach for determining a transportation problem's initial basic feasible solution (IBFS) utilising penalties. The MODI technique (Modified Distribution) is used to improve the IBFS even further. The problem was solved both manually and with the Python Programming Language. From Table 1 we can say that There are a total of 5 Supply Constraints. and There are a total of 7 demand constraints. Here, Total Demand = 5750 is more than the entire supply =4500. As a result, we add a fake supply constraint with a 0-unit cost and allocation 1250, as well as the updated supply constraint. Table (2) is given below.

Table 2: Modified Table

	Hongkong	Sydney	Amsterdam	Moscow	Toronto	Bangkok	Lisbon	Supply
Delhi	116	31	71	19	49	47	48	1200
Mumbai	35	12	108	70	49	89	90	900
Chandigarh	54	14	65	19	105	59	68	900
Hyderabad	74	109	58	48	39	51	20	800
Ahmedabad	100	102	90	30	20	58	68	700
S_{dummy}	0	0	0	0	0	0	0	1250
Demand	550	930	1250	620	900	800	700	

Now we will calculate the row and column penalties which is given in Table (3).

Table 3: Penalty Table

	Hong Kong	Sydney	Amsterdam	Moscow	Toronto	Bangkok	Lisbon	Supply	Row penalty
Delhi	116	31	71	19	49	47	48	1200	
Mumbai	35	12	108	70	49	89	90	900	12=31-19
Chandigarh	54	14	65	19	105	59	68	900	23=35-12
Hyderabad	74	109	58	48	39	51	20	800	5=19-4
Ahmedabad	100	102	90	30	20	58	68	700	19=39-20
S_{dummy}	0	0	0	0	0	0	0	1250	10=30-20
Demand	550	930	1250	620	900	800	700		0-0=0
Column penalty	35-35=0	12-12=0	58-58=0	19-19=0	20-20=0	47-47=0	20-20=0		

The maximum Penalty is 58 which occurs at Amsterdam, the minimum C_{ij} in this column is 0, Maximum amount for this cell min (1250,1250) =1250, It satisfy supply of S_{dummy} and demand of Amsterdam.

The reduced Table (4) is given below:

Table 4: Reduced Table

	Hongkong	Sydney	Amsterdam	Moscow	Toronto	Bangkok	Lisbon	Supply	Row penalty
Delhi	116	31	71	19	49	47	48	1200	12=31-19
Mumbai	35	12	108	70	49	89	90	900	23=35-12
Chandigarh	54	14	65	19	105	59	68	900	5=19-4
Hyderabad	74	109	58	48	39	51	20	800	19=39-20
Ahmedabad	100	102	90	30	20	58	68	700	10=30-20
S_{dummy}	0	0	0(1250)	0	0	0	0	0	-
Demand	550	930	0	620	900	800	700		-
Column penalty	19=54-35	2=14-12	-	0=19-19	19=39-20	4=51-47	28=48-20		

On applying the same process given in [14], we will get the initial basic feasible solution which is given in Table (5):

Table 5: Initial Basic Feasible Solution

	Hongkong	Sydney	Amsterdam	Moscow	Toronto	Bangkok	Lisbon	Supply
Delhi	116	31	71	19(300)	49(100)	47(800)	48	1200
Mumbai	35	12(900)	108	70	49	89	90	900
Chandigarh	54(550)	14(30)	65	19(320)	105	59	68	900
Hyderabad	74	109	58(d)	48	39(100)	51	20(700)	800
Ahmedabad	100	102	90	30	20(700)	58	68	700
S_{dummy}	0	0	0(1250)	0	0	0	0	1250
Demand	550	930	1250	620	900	800	700	

Optimality Test using Modi's Method:

The allocation table is given in Table (6):

Table 6: Allocation Table

	Hongkong	Sydney	Amsterdam	Moscow	Toronto	Bangkok	Lisbon	Supply
Delhi	116	31	71	19(300)	49(100)	47(800)	48	1200
Mumbai	35	12(900)	108	70	49	89	90	900
Chandigarh	54(550)	14(30)	65	19(320)	105	59	68	900
Hyderabad	74	109	58(d)	48	39(100)	51	20(700)	800
Ahmedabad	100	102	90	30	20(700)	58	68	700
S_{dummy}	0	0	0(1250)	0	0	0	0	1250
Demand	550	930	1250	620	900	800	700	

Using the steps which is given in [14], we get the final matrix which is given in Table (7).

Table 7: Final Table

	Hongkong	Sydney	Amsterdam	Moscow	Toronto	Bangkok	Lisbon	Supply
Delhi	116	31	71	19(300)	49(100)	47(800)	48	1200
Mumbai	35(550)	12(350)	108	70	49	89	90	900
Chandigarh	54	14(580)	65(d)	19(320)	105	59	68	900
Hyderabad	74	109	58	48	39(100)	51	20(700)	800
Ahmedabad	100	102	90	30	20(700)	58	68	700
S_{dummy}	0	0	0(1250)	0	0	0	0	1250
Demand	550	930	1250	620	900	800	700	

The pseudo python programming code is constructed for larger data sets which is given in [14].

Case 2: Crew Scheduling

We must identify an assigned task of the requisite technical crew members to After producing all eligible flight services, these flight services cover each flight segment at least once, such that each scheduled flight segment is conducted by a qualified technical crew The goal of this strategy is to find a workable low-cost project. such that assigns each crew to flights with the lowest possible cost.

The crew scheduling problem is usually divided into two subproblems, the crew pairing problem and the crew assignment problem, which are addressed in below:

1) Crew Pairing Problem

The problem produces a pair, which are low-cost, multi-day work schedules. Airlines explain how flight legs can be joined to build schedules that are feasible. The number of feasible pairings measures is estimated using work-rule restrictions such as the maximum number of hours worked by crews in a day and the least number of hours of rest between work periods.

2) Crew Assignment Problem

This challenge combines these pairs into month-long crew plans that are both fair and efficient., which are referred to as roasters, and are assigned to individual crew members. Rostering is the process of creating and assigning schedules to specific individuals based on their specific requests. [3,4]

Solving Crew Scheduling Problems

The Hungarian Method is used to solve the crew scheduling problem.[7]

Air India has two major domestic hubs: I. G International Airport in Delhi and C.S.M International Airport in Mumbai (Mumbai). These two cities are critical to Air India's domestic operations and have a significant impact on its scheduling. According to Air India's current timetables, there are 16 flights scheduled between Delhi and Mumbai. There were ten direct schedules from both cities. For the existing schedule, assuming all flights and crew are stationed in Delhi, the rest duration for crew members in Mumbai was found to be 6 hours [5], as shown in Figure 1.

Delhi-Mumbai			Mumbai-Delhi		
Flight No	Departure	Arrival	Flight No	Departure	Arrival
a	06.00	12.00	1	05.30	11.30
b	07.30	13.30	2	09.00	15.00
c	11.30	17.30	3	15.00	21.00
d	19.00	01.00	4	18.30	00.30
e	00.30	06.30	5	00.00	06.00
f	06.30	11.30	6	06.00	11.30
g	08.30	13.00	7	09.30	15.30
h	11.00	17.00	8	15.30	21.00
i	19.30	01.30	9	19.30	00.30
j	12.30	07.30	10	00.30	06.00

Figure 1

The complete steps to solve the crew scheduling Problem can be seen in [14].

To determine optimal assignments, first we calculate layover times from the above time table which is given in [14].

Using the layover time, we obtained the matrix which are crew based at Delhi & crew based at Mumbai are given in Table [9] & Table [10].

Table 9: Crew Based at Delhi

	1	2	3	4	5	6	7	8	9	10
A	17.5	21	27	6.5	12	18	21.5	27.5	7.5	12.5
B	16	19.5	25.5	5	10.5	16.5	20	26	6	11
C	12	15.5	21.5	25	6.5	12.5	16	22	26	7
D	4.5	8	14	17.5	23	5	8.5	14.5	18.5	23.5
E	23	26.5	8.5	12	17.5	23.5	27	9	13	18
F	18	21.5	27.5	7	12.5	18.5	22	28	8	13
G	16.5	20	26	5.5	11	17	20.5	26.5	6.5	11.5
H	12.5	16	22	25.5	7	13	16.5	22.5	26.5	7.5
I	28	7.5	13.5	17	22.5	4.5	8	14	18	23
J	22	25.5	7.5	11	16.5	22.5	26	8	12	17

Table 10: Crew Based at Mumbai

	1	2	3	4	5	6	7	8	9	10
A	18.5	15	9	5.5	24	18.5	14.5	9	5.5	24
B	20	16.5	10.5	7	25.5	20	16	10.5	7	25.5
C	24	20.5	14.5	11	5.5	24	20	14.5	11	5.5
D	7.5	28	22	18.5	13	7.5	27.5	22	18.5	13
E	13	9.5	27.5	24	18.5	13	9	27.5	24	18.5
F	19	15.5	9.5	6	24.5	19	15	9.5	6	24.5
G	21	17.5	11.5	8	26.5	21	17	11.5	8	26.5
H	23.5	20	14	10.5	5	23.5	19.5	14	10.5	5
I	8	4.5	22.5	19	13.5	8	28	22.5	19	13.5
J	25	21.5	15.5	12	6.5	25	21	15.5	12	6.5

The composite layover time matrix is created by selecting the smallest element from the two matched elements in Table (9) and (10). The Crew Based is Mumbai if the (*) present; else The Crew is Based in Delhi in Table (11).

Table 11

	1	2	3	4	5	6	7	8	9	10
a	17.5	15*	9*	5.5*	12	18	14.5*	9*	5.5*	12.5
b	16	16.5*	10.5*	5	10.5	16.5	16*	10.5*	6	11
c	12	15.5	14.5*	11*	5.5*	12.5	16	14.5*	11*	5.5*
d	4.5	8	14	17.5	13*	5	8.5	14.5	18.5*	13*
e	13*	9.5*	8.5	12	17.5	13*	9*	9	13	18
f	18	15.5*	9.5*	6*	12.5	18.5	15*	9.5*	6*	13
g	16.5	15.5*	11.5*	5.5	11	17	17*	11.5*	6.5	11.5
h	12.5	16	14*	10.5*	5*	13	16.5	14*	10.5*	5*
i	8*	4.5*	13.5	17	13.5*	4.5	8	14	18	13.5*
j	22	21.5*	7.5	11	6.5*	22.5	21*	8	12*	6.5*

Now, Solve the above problem using Hungarian Method and the complete steps are given in [14].and the final matrix is given in Table (12).

Table 12: Optimal Assignments

	1	2	3	4	5	6	7	8	9	10
a	3	0	0	1	4	3	0	[0]	0	4.5
b	1	1	1	0	2	1	1	1	[0]	2.5
c	0	3	8	9	[0]	0	4	8	8	0
d	[0]	3	15	23	15	0	4	15.5	23	15
e	4	0	5	13	15	3.5	[0]	5.5	13	15.5
f	3	[0]	0	1	4	3	0	0	0	4.5
g	1	1.5	1.5	[0]	2	1	1.5	1.5	1	2.5
h	1	4	8	9	0	1	5	8	8	[0]
i	4	0	15	23	16	[0]	4	15.5	23	16
j	9	8	[0]	8	0	9	8	0.5	8	0

The pseudo python programming code is constructed for larger data sets which is given in [14].

3.2 Case 3: Air India-Transporting numerous items in a one plane

The problem consists of a transportation network, as well as a collection of diverse commodities, such as cargo goods, passenger classes, and so on, all of which are distinguished by their physical characteristics. When different types of goods are transmitted across the same network, which will benefit the airline business by lowering costs and increasing revenues, as well as allowing them to transport multiple items at once, shortening travel time. This process will take longer if they opt to deliver cargo in one plane and passengers in different flights.

Let us define the model in following manner. The first is the I. G International Airport in Delhi, and the second is the C. S.M International Airport in Mumbai. From Delhi to Mumbai, we must transport both cargo goods and passengers.

There are many destinations between Delhi and Mumbai, thus we must choose the quickest route between them so that cargo and people are transported in less time, which is beneficial to the airline sector. Let’s call I.G International Airport 1 and C. S. M International Airport 9, respectively. and the different notations used to express different locations between them are as follows:

2 = Jaipur ,3 = Kota ,4 = Ahmedabad, 5 = Indore, 6 = Vadodara, 7 = Surat,8 = Nasik.

Below is the route map with all of the distances between each place for the above problem.

The main aim is to find the shortest distance between Airport 1 and Airport 9 such that the cost is minimized and the route map is given in Figure 2.

We have assumed distance in hundreds

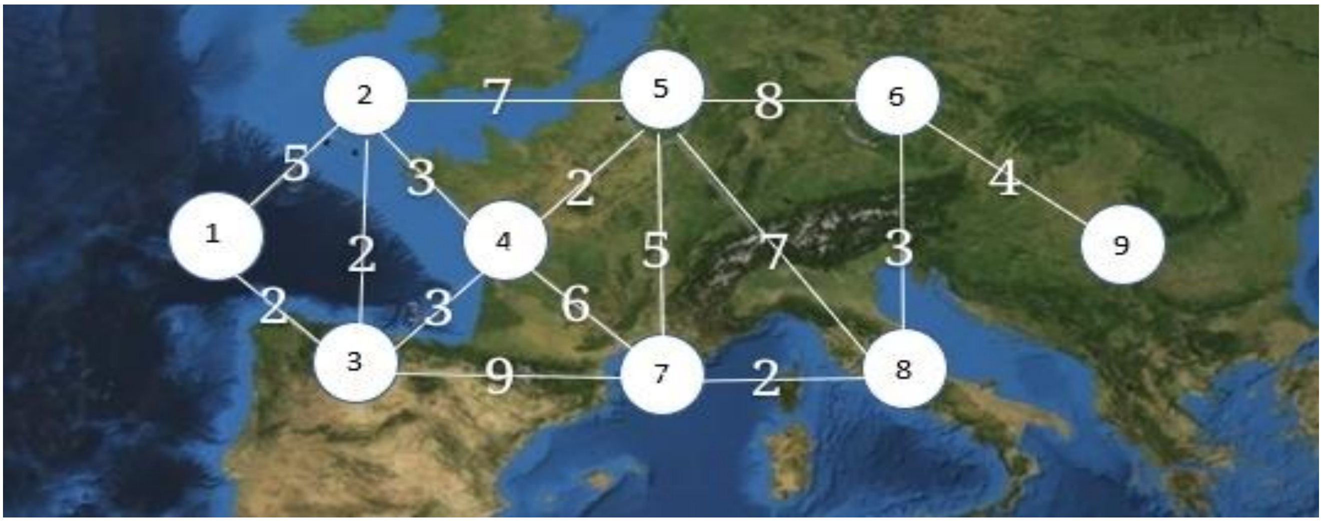

Figure 2: Route Map

What if you were given a graph of nodes in which each node is connected to a number of other nodes at varying distances? What is the shortest path to every other node in the network if you start from one of the nodes in the graph? Dijkstra's Algorithm is a simple algorithm that is used to find the shortest distance [8], [9], or path, between a beginning node to a target node in a weighted network. This technique is being used to find the quickest way between two airports.

Dijkstra's algorithm is an iterative algorithm that finds the shortest path from a single beginning node to all other nodes in a graph [9], [10]. This algorithm is used to solve the model, and the final result is shown below in Figure 3.

	A	B	C	D	E	F	G	H	I
A	0	∞	∞	∞	∞	∞	∞	∞	∞
B	0	5	2	∞	∞	∞	∞	∞	∞
C	0	4	2	∞	∞	∞	11	∞	∞
D	0	4	2	7	11	∞	11	∞	∞
E	0	4	2	7	9	∞	11	∞	∞
F	0	4	2	7	9	17	11	16	∞
G	0	4	2	7	9	17	11	13	∞
H	0	4	2	7	9	16	11	13	∞
I	0	4	2	7	9	16	11	13	∞

Figure 3: Dijkstra's Solution

The pseudo python programming code is constructed for larger data sets which is given in Figure 4.

```
dijkstra algorithm

function stra(matrix,unvsett,shortdistance,node,source):
    s = node
    print(node)
    unvsett.remove(s)//remove current node from unvisited set
    print(unvsett)//print unvisited set

for i in unvsett:// running the loop in unvisited set
if  matrix[s][i] != 0:// checking if s and i are connected in the graph
if  shortdistance[s]+matrix[s][i] < matrix[source][i] ://if distance from the source(edge length) is greater than distance
(through path from s(current node)
shortdistance[i] = shortdistance[s]+matrix[s][i];//update the shortdistance array with the newly found short distance
elif shortdistance[s]+matrix[s][i] > matrix[source][i] & matrix[source][i] == 0://if the node i is not connected to the source node
                shortdistance[i] = shortdistance[s]+matrix[s][i];//updating shortest distance with only possible path distance
    print(shortdistance)
    if len(unvsett) > 0://checking  if unvisited set is empty
        k = len(unvsett)

        l = 0;
        for i in range(k):// finding the node with least possible shortdistance from the source
            if shortdistance[unvsett[l]] > shortdistance[unvsett[i]] & shortdistance[unvsett[i]] > 0:
                l = i;
        node = unvsett[l];//the node with least distance from source

stra(matrix,unvsett,shortdistance,node,source);//recalling the function with new updated unvisited set and new current node
if unvisited set is not empty
    else:
        print(shortdistance);//printing the shortdistance array if all the nodes are visited

Main function
m = [[0, 4, 0, 0, 0, 0, 0, 8, 0], //// matrix
        [4, 0, 8, 0, 0, 0, 0, 11, 0],
        [0, 8, 0, 7, 0, 4, 0, 0, 2],
        [0, 0, 7, 0, 9, 14, 0, 0, 0],
        [0, 0, 0, 9, 0, 10, 0, 0, 0],
        [0, 0, 4, 14, 10, 0, 2, 0, 0],
        [0, 0, 0, 0, 0, 2, 0, 1, 6],
        [8, 11, 0, 0, 0, 0, 1, 0, 7],
        [0, 0, 2, 0, 0, 0, 6, 7, 0]
        ];
u = [0,1,2,3,4,5,6,7,8];//nodes
s = 0;//source node
sd = m[s];//edge set of source node
stra(m,u,sd,0,0)//function
```

Figure 4: Algorithm 2

3.3 Case4: Aircraft Maintenance Problem

As company levels of operational performance, complexity, and expense continue to rise, modern aircraft turbine engines need new and increasingly severe management, operations, and maintenance difficulties. Repair and overhaul of aircraft gas turbine engines, Repairing and overhauling gas turbine engines, rebuilding gas turbine engines, balancing components and assemblies, testing and troubleshooting gas turbine engines, and inspecting gas turbine engine components and assemblies are all jobs for technicians [11,12,13]. As a consequence, depending on business standards, the scope of the examination may be separated into multi-level abilities, all technicians are employed needed to do only a subset of the duties stated in the analysis. This aviation engine specialist's responsibilities include troubleshooting and examining the components and assemblies of gas turbine engines. This can entail dismantling and rebuilding engines, as well as balancing components in these complex [5], contemporary engines. Engine overhauls are carried out in specialised shops where engines are dismantled, issues are detected, and Components of the engine are cleaned, repaired, or rebuilt. After that, the engine is reassembled and examined.

The Performance and tolerances standards for aviation gas turbine engines are extremely tight, necessitating the use of highly qualified experts with excellent mechanical skills. Workers in this position must be. Materials management data bases have been built in engine repair facilities, and keyboarding abilities are becoming increasingly crucial.

Any maintenance job entails a variety of tasks. In this case, A model of aircraft maintenance is built, which aids in the prioritization of the most important aircraft maintenance activity. Here, actions are represented by nodes in a network. The activities and their durations are presented inside the node when showing activities on an Activity-On-Node or CPM network [11,12,13]. The network "critical route," which consists of the sequence of project operations that determines the lowest necessary project time, is computed using the CPM approach Consequently [11].

All of the duration are in the project outlined in Table: 13

Table 13: CPM Duration

No.	Code	Activity Description	Precedence Activity	Activity time
1	A	Check stand by pump	-	Around 32 man-hours
2	B	Calibrate all ganges	A	Around 80 man-hours
3	C	Disassemble Pump cover and remove rotor	A	Around 8 man-hours
4	D	Disassemble Turbine cover and remove	A	Around 16 man-hours
5	E	rotor	B	Around 32 man-hours
6	F	Clean all ganges and line	E	Around 16 man-hours
7	G	Replace ganges	C	Around 28 and 40 man-hours
8	H	Repair lubrication system	C	Around 256 and 352manhours
9	I	Rebuild impeller	C	Around 32 man-hours
10	J	Clear pump casting	G	Around 16 man-hours
11	K	Fix pump bearings	H	Around 32 man-hours
12	L	Balance impeller	I, J, K	Around 16 man-hours
13	M	Reinstall impeller	D	Around 260 and 400manhours
14	N	Rebuild turbine rotor	D	Around 16 man-hours
15	O	Check turbine bearings	M	Around 64 man-hours
16	P	Balance turbine rotor	N	Around 16 man-hours
17	Q	Fix turbine bearings	O, P	Around 16 and 28 man-hours
18	R	Fix turbine rotor	Q	Around 20 and 32 man-hours
19	S	Fix turbine cover	R	Around 12 and 18 man-hours
20	T	Text components	R	Around 16 man-hours
21	U	Check clearance	L	Around 16 man-hours
22	V	Fix pump bearings	U	Around 18 and 24 man-hours
23	W	Fix pump cover	V	Around 16 man-hours
24	X	Install shaft packing	S, T, W, F	Around 48 and 72 man-hours
		Final test		

Float, Free float, independent float is given in Table 14

Table 14: Float, Free Float & Independent Float

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
-	A	A	A	B	E	C	C	C	G	H	I,J,K	D	D	M	N	O,P	Q	R	R	L	U	V	S,T,W,F
32	80	8	16	32	15	40	32	90	70	40	70	30	60	50	30	50	40	60	50	40	50	10	60

Edge and its prec eded, succeed node are given in Figure 5.

Edge	Node1 → Node2
A	1 → 2
B	2 → 3
C	2 → 4
D	2 → 5
E	3 → 6
G	4 → 8
H	4 → 9
I	4 → 10
M	5 → 12
N	5 → 13
F	6 → 7
X	7 → 20
J	8 → 10
K	9 → 10
L	10 → 11
U	11 → 18
O	12 → 14
P	13 → 14
Q	14 → 15
R	15 → 16
T	16 → 7
S	16 → 17
d	17 → 7
V	18 → 19
W	19 → 7

Figure 5

The project's network diagram, including with activity times., is given in Figure 6.

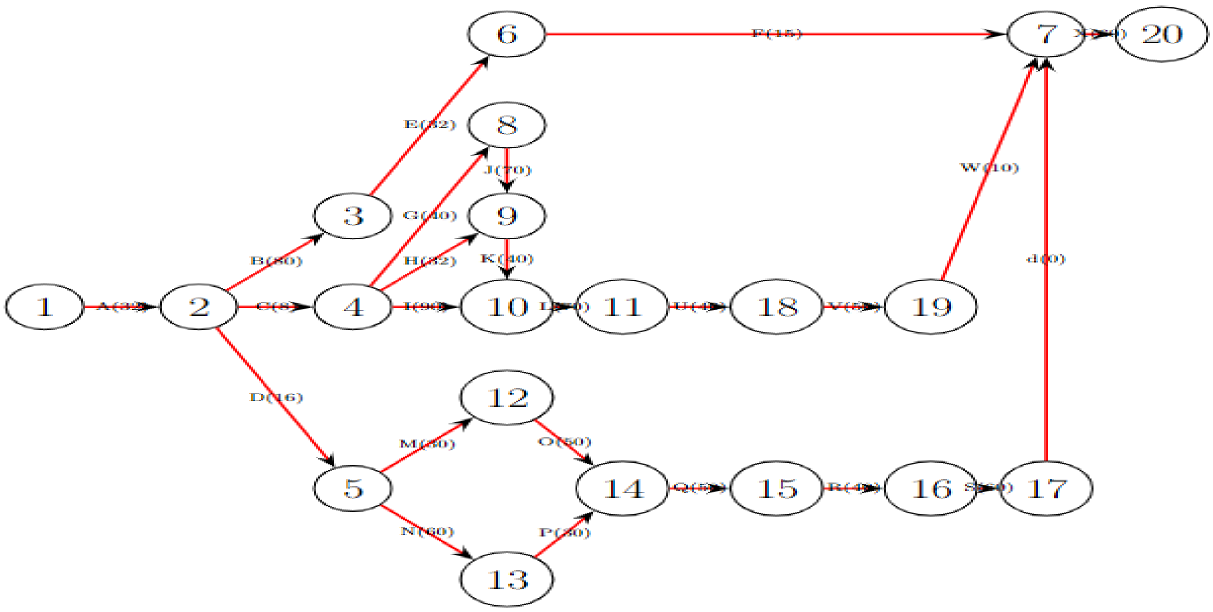

Figure 6: Network Diagram

Now solved with forward pass method and backward pass method and we will get the final result. Calculations of total float, free float, and independent float for each non-critical activity are shown in figure 7:

Activity (i,j) (1)	Duration (t_{ij}) (2)	Earliest time start (E_i) (3)	(E_j) (4)	(L_i) (5)	Latest time finish (L_j) (6)	Earliest time finish $(E_i + t_{ij})$ (7)=(3)+(2)	latest time start $(L_j - t_{ij})$ (8)=(6)-(2)	total float $(L_j - t_{ij}) - E_i$ (9)=(8)-(3)	Free float $(E_j - E_i) - t_{ij}$ (10)=(4)-(3)-(2)	Independent Float $(E_j - L_i) - t_{ij}$ (11)=(4)-(5)-(2)
2-3	80	32	112	32	273	112	193	161	0	0
2-5	16	32	48	32	80	48	64	32	0	0
3-6	32	112	144	273	305	144	273	161	0	-161
4-9	32	40	72	40	110	72	78	38	0	0
4-10	90	40	150	40	150	130	60	20	20	20
5-12	30	48	78	80	120	78	90	42	0	-32
5-13	60	48	108	80	140	108	80	32	0	-32
6-7	15	144	320	305	320	159	305	161	161	0
9-10	40	72	150	110	150	112	110	38	38	0
12-14	50	78	138	120	170	128	120	42	10	-32
13-14	30	108	138	140	170	138	140	32	0	-32
14-15	50	138	188	170	220	188	170	32	0	-32
15-16	40	188	228	220	260	228	220	32	0	-32
16-7	50	228	320	260	320	278	270	42	42	10
16-17	60	228	288	260	320	288	260	32	0	-32
17-7	0	288	320	320	320	288	320	32	32	0

Figure 7: Final Result

The complete pseudo python programming code is constructed for larger data sets which is given in [14].

```
DEFINE FUNCTION analysis(rows):
    FOR eachTask IN rows.keys():
        SET rows[eachTask]['ES'] TO 0
        SET rows[eachTask]['EF'] TO 0
        SET rows[eachTask]['LS'] TO 0
        SET rows[eachTask]['LF'] TO 0
        SET rows[eachTask]['float'] TO 0
        SET rows[eachTask]['isCritical'] TO False
    FOR taskFW IN rows:
        IF '-1' IN rows[taskFW]['$d']:
            SET rows[taskFW]['ES'] TO 1
            SET rows[taskFW]['EF'] TO (rows[taskFW]['duration'])
        ELSE:
            FOR k IN rows.keys():
                FOR eachDependency IN rows[k]['$d']:
                    IF (eachDependency != '-1' and len(
                            rows[k]['$d']) EQUALS 1):
                        SET rows[k]['ES'] TO int(rows['task' + eachDependency]['EF']) + 1
                        SET rows[k]['EF'] TO int(rows[k]['ES']) + int(rows[k]['duration']) - 1
                    ELSEIF eachDependency != '-1':
                        IF int(rows['task' + eachDependency]['EF']) > int(rows[k]['ES']):
                            SET rows[k]['ES'] TO int(rows['task' + eachDependency]['EF']) + 1
                            SET rows[k]['EF'] TO int(rows[k]['ES']) + int(rows[k]['duration']) - 1
    SET tempList TO list()
    FOR element IN rows.keys():
```

Results and Discussion

4.1 Result for case 1:

As a result, we've arrived at the best optimal solution.

Table 15: optimal Table

	Hongkong	Sydney	Amsterdam	Moscow	Toronto	Bangkok	Lisbon	Supply
Delhi	116	31	71	19(300)	49(100)	47(800)	48	1200
Mumbai	35(550)	12(350)	108	70	49	89	90	900
Chandigarh	54	14(580)	65(d)	19(320)	105	59	68	900
Hyderabad	74	109	58	48	39(100)	51	20(700)	800
Ahmedabad	100	102	90	30	20(700)	58	68	700
S_{dummy}	0	0	0(1250)	0	0	0	0	1250
Demand	550	930	1250	620	900	800	700	

The lowest possible overall transportation cost:

19×300+49×100+47× 800+35×550+12×350+14×580+19×320+39×100+20×700+0×1250

=**117750.**

4.2 Result for case 2:

The most optimum solution:

Table 16: optimal Table

Work	A	B	C	d	E	f	g	h	i	j	
Job	8	9	5	1	7	2	4	10	6	3	Total
Cost	9	6	5.5	4.5	9	15.5	5.5	5	4.5	7.5	72

4.3 Result for case 3:

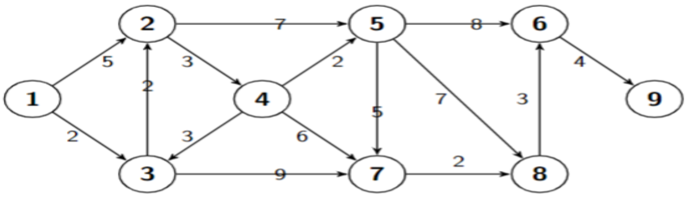

Figure 8: shortest route

The Shortest Path Is: 1→ 3→ 7 → 8→ 6 → 9

The Shortest Path Is From: Delhi →Kota → Surat→ Nasik → Vadodara→ Mumbai

4.4 Result for case 4:

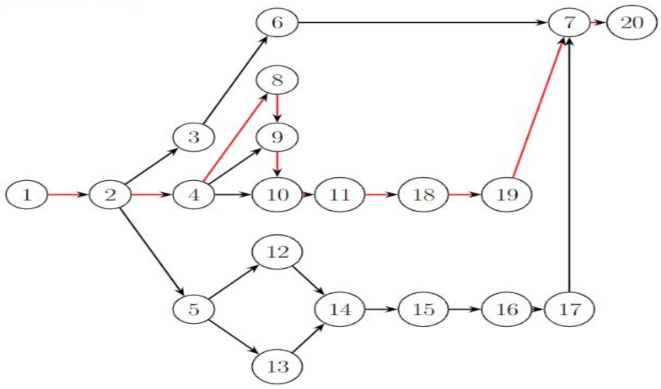

Figure 9:

The Critical path is: 1→ 2→ 4→ 8→ 10→ 11→ 18→ 19→ 7→ 20

The Critical Activities are: A, C, G, J, L, U, V, W, X

The total time is 380

CONCLUSION

Approaches based on operations research can be quite beneficial in terms of route optimization and resource utilisation. It provides a simple and practical solution for better decision-making, as well as time, resource, and cost savings for businesses. We employed the Hungarian approach of assignment algorithm to reroute flights between Mumbai and Delhi in order to increase crew rest time for a certain airline operator in this study.

As a result, OR methods can be particularly useful in determining flight times and routes in order to save operational costs. After analysing the aircraft maintenance activities of the airline using Critical Path Analysis, it can be concluded that CPM is a valuable tool for reducing time elapsed and increasing flying hours, which in turn increases the airline's profits, and we also solve the airline problem with various cases. Any aviation industry can benefit from the algorithms we mentioned above by saving time and money while increasing profits

ACKNOWLEDGEMENT

I would like to thank to my supervisor Prof. Ln Das for his assistance and sharing their pearls of wisdom with me during the course of this research.

REFERENCES

[1] HARVEY H. SHORE. (2007). The Transportation Problem and the Vogel Approximation Method. Decision Sciences 13(4), (441–457). DOI: 10.1111/j.1540-5915.1970.tb00792.x

[2] Krishan Kasad,Kavisha Doshi, Kavya Kaushik, Khushboo Balani, KrishNathany(2021) Application of Operations Research in the Airline Industry. International Journal of Scientific Research and Engineering Development 4(5), (827-837). Retrieved March 2, 2022, from: http://www.ijsred.com/volume4/issue5/IJSRED-V4I5P86.pdf.

[3] Ira Gershkoff.(Jul.-Aug.,1989).Optimizing Flight Crew Schedules.19(4)(29-43).Retrieved March 5,2022,from:http://www.jstor.com/stable/25061246

[4] Cynthia Barnhart, Peter Belobaba and Amedeo R. Odoni.(2003).Application of Operations Research in the Air Transport Industry.Transportation Science.37(4).Issue on Aviation Operations Research: Commemorating100 Years of Aviation (November 2003),368-391.Retrieved March 8 ,2022, from: https://www.jstor.org/stable/25769164?seq=1&cid=pdf , Varun, Mittal. Vidisha, Diwan. Anand, Baid.(2017). A Study on The Application of

[5] Operational Research in The Airline Industry. International Journal of Innovative Science and Research Technology. 2(10), (228-241), Retrieved March10, 2022, from: https://ijisrt.com/wp-content/uploads/2017/10/A-STUDY-ON-THE-APPLICATION-OF-OPERATIONS-RESEARCH-IN-THE-AIRLINE-INDUSTRY-1-ilovepdf-compressed.pdf

[6] F. M. Zeghal&M. Minoux. (2004).Modeling and solving a Crew Assignment Problem in air transportation. European Journal of Operational Research 175 (2006), 187–209.Doi:10.1016/j.ejor.2004.11.028

[7] Sonakshi Khanna, Trusha Dholakiya, Suraj Prakash, Shubhang Gupta, Udit Jain. (2021) Application of the Hungarian Assignment Method in the Aviation Industry. International Journal of Innovative Science and Research Technology.6(10), (300-305). Retrieved March 18, 2022, from: https://ijisrt.com/assets/upload/files/IJISRT21OCT220.pdf

[8] Solai Rani P. (2021). Application of Graph Theory in Air-Transportation Network. J PurAppl Math. 2021; 5(1), (1-4). Retrieved March 22, 2022, from:https://www.pulsus.com/scholarly-articles/application-of-graph-theory-in-airtransportation-network.pdf.

[9] M. A. Javaid, (2013). Understanding Dijkstra algorithm, SSRN Electronic Journal, (1-27) . Retrieved March 28, 2022 from: Doi: 10.2139/ssrn.2340905

[10] Israa Ezzat Salem, Maad M. Mijwil, Alaa Wagih, Marwa M. Ismaeel. (2022), Flight-schedule using Dijkstra's algorithm with comparison of routes findings. International Journal of Electrical and Computer Engineering 12(2). (1675-1682). Retrieved April 1, 2022, from: DOI: 10.11591/ijece.v12i2.pp1675-1682

[11] Indrawaty.Y. Marit, Ellysa Nursanti, Prima Vitasari. (2022). Critical Path Method to Accelerate Automotive Maintenance Duration. INTERNATIONAL JOURNAL OF SCIENTIFIC & TECHNOLOGY RESEARCH. 9(3), (6777-6782). Retrieved April 4, 2022, from: https://www.ijstr.org/final-print/mar2020/Critical-Path-Method-To-Accelerate-Automotive-Maintenance-Duration.pdf.

[12] Tzeu-Chen Han, Cheng-Chi Chung, Gin-Shuh Liang. (2006). Application of fuzzy critical path method to airport's cargo ground operation systems. Journal of Marine Science and Technology,14(3), (139-146). Retrieved April 8, 2022 from: • DOI: 10.51400/2709-6998.2067

[13] Junqueira, V.S.V. Nagano, M.S. Miyata, H. H. (2018) Procedure Structuring for Programming Aircraft Maintenance Activities. Revista de Gestão.27(2), (2-20). Retrieved April 9, 2022. From: www.emeraldinsight.com/2177-8736.htm, 2018 DOI: 10.1108/REGE-02-2018-0026

[14] Vedika Dixit, Das.L (2022) APPLICATIONS OF OPERATIONAL RESEARCH IN AIRLINE MANAGEMENTAND SCHEDULING. Retrieved April, 2022 from: https://drive.google.com/file/d/14uDEadOwo24Fk2tNeLQsM6WwcQqV0fy5/view?usp=sharing