Northern Illinois University

ISYE 671 LINEAR PROGRAMMING AND NETWORK FLOWS

PROJECT

Submitted by

Bhaveshkumar Chandrasekaran	**Z2006107**
Md Abu Naeem Khan	**Z1985093**

College of Engineering and Engineering Technology

Department of Industrial and Systems Engineering

Fall 2023

Table of Contents

List of Tables

List of Figures

1.Introduction:

In the transportation industry, a company's ability to succeed depends on its supply chain management. The cost, duration, and distance of transportation are factors in a supply chain scheduling issue. On the other hand, tardiness is punished in this paper for failing to arrive at the assignment location by the deadline. Modern supply chains are complex and navigating them requires a careful balancing act between accuracy and efficiency. The Supply Chain Scheduling, an essential component of logistics management, is at the centre of this arrangement. This field deals with the strategic planning and synchronization of production procedures, logistics for transportation, and delivery schedules to guarantee the smooth transition of goods from the point of production to the customers.

The most distinctive feature of this paper is its optimization to minimize the overall cost, which includes transportation expenses, fixed car costs, and delivery delays caused by the inability to reach the destination, which distinguishes the problem.

2. Problem Statement:

In the present study [1], we examine an instance in which an individual manufacturer (origin) engages with several clients via a transportation network, symbolized by a collection of arcs. Every time a customer places an order, the manufacturer has a task on their hands. The difficulty lies in efficiently scheduling these jobs to meet deadlines while taking transportation and processing times into account. The optimization problem gains a time-sensitive dimension when it comes to the jobs, as tardiness carries penalties.

The introduction of vehicles complicates the situation even more because each one has a fixed operational cost and a limited capacity. The goal is to reduce the total cost, which includes fixed vehicle operating costs, travel expenses, and fines for late deliveries. This formulation of the problem, which incorporates scheduling complexities and is inspired by the Vehicle Routing Problem (VRP), emphasizing its computational complexity.

3. Problem Formulation:

Indices

i, j	Jobs
k	Vehicles
O	Origin

Sets

I	Set of jobs
K	Set of vehicles
N	I U $\{O\}$

Parameters

C_{ij}	Travel cost between customer of job i and customer of job j
t_{ij}	Travel time between customer of job i and customer of job j
P_i	Processing time associated to job i
w_i	Penalty for unit delivery tardiness of job i
d_i	Due date associated to job i
F	Fixed cost associated to use of a vehicle
Q	Capacity of each vehicle

Decision variables

X^k_{ij}	Binary variable which takes 1 if vehicle k does drive from customer of job i to customer of job j, and 0 otherwise
Y_k	Binary variable which takes 1 if vehicle k is used and 0 otherwise
A_{ij}	Binary variable which takes 1 if job i is processed before job j, and 0 otherwise
S_k	Start time of vehicle k
T_i	Delivery delay of job i
D_i	Delivery time of job i

4. Mathematical programming

To find the optimal solution compatible with the designed problem, a mixed integer linear programming is proposed, as follows:

$$\boldsymbol{Minimize} \sum_{(i,j)\in A} \sum_{k\in K} C_{ij}\, X_{ij}^{k} + F \sum_{k\in K} Y_k + \sum_{k\in K} w_i\, T_i$$

$$\sum_{k\in K} \sum_{j\in J, j\neq i} X_{ij}^{k} = 1 \qquad \forall i \in I$$

$$\sum_{i\in N} \sum_{j\in N, j\neq i} X_{ij}^{k} \leq (Q+1)\, Y_k \qquad \forall k \in K$$

$$\sum_{j\in N} X_{oj}^{k} = Y_k \qquad \forall k \in K$$

$$\sum_{i\in N} X_{ih}^{k} = \sum_{j\in N} X_{hj}^{k} \qquad \forall h \in N, \forall k \in K$$

$$A_{ij} + A_{ji} = 1 \qquad \forall i, j \in N, i \neq j$$

$$A_{ij} + A_{jr} + A_{ri} \geq 1 \qquad \forall i, j, r \in N, i \neq j \neq r$$

$$S_k \geq \sum_{i\in N} \sum_{j\in N, i\neq j} (p_i\, A_{ij}) + p_j - M(1 - \sum_{j\in N, j\neq i} X_{ij}^{k}) \qquad \forall k \in K, \forall j \in N$$

$$D_j - S_k \geq t_{oj} - M\left(1 - X_{oj}^{k}\right) \qquad \forall k \in K, \forall j \in N$$

$$D_j - D_i \geq t_{ij} - M\left(1 - \sum_{k\in K} X_{ij}^{k}\right) \qquad \forall i, j \in N$$

$$T_i \geq D_i - d_i \qquad \forall i, j \in I$$

$$X_{ij}^{k} \in \{0, 1\} \qquad \forall i, j \in I, \forall k \in K$$

$$Y_k \in \{0, 1\} \qquad \forall k \in k$$

$$A_{ij} \in \{0, 1\} \qquad \forall i, j \in I, \forall k \in K$$

$$S_k \geq 0 \qquad \forall k \in K$$

$$D_i \geq 0 \qquad \forall i \in I$$

$$T_i \geq 0 \qquad \forall i \in I$$

4.1 Constraint 1:

This constraint ensures that every job is served by exactly one vehicle. The sum over vehicles and nodes ensures that for each job i, there is only one vehicle that travels from any node j (excluding the job's own node) to serve that job.

4.2 Constraint 2:

This constraint enforces the capacity constraint for each vehicle. It ensures that the total load (sum of jobs) on each vehicle does not exceed its capacity. The binary variable Y_k is used to indicate whether vehicle k is used or not.

4.3 Constraint 3:

This constraint ensures that a vehicle is considered "used" (binary variable Y_k is 1) if it makes any trips. It sums over all nodes j for vehicle k starting from the depot.

4.4 Constraint 4:

This is a flow conservation constraint for each job and vehicle. It ensures that the flow of a vehicle in and out of a node is balanced. The sum over nodes i represents the flow into node h, and the sum over nodes j represents the flow out of node h.

4.5 Constraint 5:

This constraint enforces the sequencing of jobs. It ensures that if job i precedes job j in the sequence (binary variable A_{ij} is 1), then job j cannot precede job i in the sequence.

4.6 Constraint 6:

This is a subtour elimination constraint. It prevents the formation of subtours (loops) in the sequence of jobs. It ensures that, for any three distinct jobs i, j, and r, at least one pair must be in a specific order.

4.7 Constraint 7:

This constraint determines the start time for each vehicle's route. It considers the processing times of preceding jobs and the travel time from the depot to the current node. The large constant M is used as a big-M penalty term to handle binary decision variables.

4.8 Constraint 8:

This constraint calculates the delay for each job. It ensures that the delivery time (D_j) minus the start time of the vehicle (S_k) is greater than or equal to the travel time from the depot to the job (t_{0j}). The penalty term is used to handle the binary decision variable.

4.9 Constraint 9:

This constraint enforces the delivery time for each job. It ensures that the difference between the delivery times of jobs i and j is greater than or equal to the travel time between them. The penalty term is used to handle the binary decision variable.

4.10 Constraint 10:

This constraint calculates the tardiness for each job. It ensures that the tardiness (T_i) is greater than or equal to the difference between the delivery time of a job (D_i) and its due date (d_i).

5. CPLEX Mod File & Data File:

```
range Jobs = 1..10;
range Vehicles = 1..10;
range Nodes = 0..10 ;

int c[Nodes][Nodes] = ...;  // Cost matrix
int t[Nodes][Nodes] = ...;  // Time matrix
int p[Jobs] = ...;          // Processing times
int w[Jobs] = ...;          // Penalties for tardiness
int d[Jobs] = ...;          // Due dates
int F = ...;                // Fixed cost for vehicle use
int Q[Vehicles] = ...;      // Vehicle capacities
int M = ...;                // Big M

dvar boolean x[Nodes][Nodes][Vehicles];
dvar boolean y[Vehicles];
dvar boolean A[Jobs][Jobs];
dvar float+ S[Vehicles];
dvar float+ T[Jobs];
dvar float+ D[Jobs];

minimize
  sum(i in Nodes, j in Nodes, k in Vehicles) c[i][j] * x[i][j][k] + F * sum(k in Vehicles) y[k] + sum(i in Jobs) w[i] * T[i];

// Constraint
subject to {
   forall(i in Jobs)
    sum(k in Vehicles) sum(j in Nodes : j != i) x[i][j][k] == 1; //Every job is served by exactly one vehicle
   forall(k in Vehicles)
       sum(i in Nodes) sum(j in Nodes : j != i) x[i][j][k] <= Q[k]*y[k] + y[k];  // Each vehicle's capacity is not exceeded
   forall(k in Vehicles)
    sum(j in Nodes) x[0][j][k] == y[k]; //  A vehicle is used if it makes any trips
   forall(h in Nodes, k in Vehicles)
    sum(i in Nodes) x[i][h][k] == sum(j in Nodes) x[h][j][k]; //i in nodes // Flow conservation for each job and vehicle
   forall(i in Jobs, j in Jobs : i != j) // Sequence of jobs
    A[i][j] + A[j][i] == 1;
   forall(i in Jobs, j in Jobs, r in Jobs : i != j && i != r && j != r) // Subtour elimination = prevent facing to impossible sequences of the jobs which is
     A[i][j] + A[j][r] + A[r][i] >= 1;                              // required to compute the completion time of each job
 forall(j in Nodes: j>0, k in Vehicles)
    S[k] >= sum(i in Nodes : i>0 && i != j) (p[i] * A[i][j]) + p[j] - M * (1 - sum(i in Nodes :i >0 && i != j) x[i][j][k]); // Start time for each vehicle(8)
    forall(j in Jobs, k in Vehicles)
    D[j] - S[k] >= t[0][j] - M * (1 - x[0][j][k]); // Delay for each job
  forall(i in Jobs, j in Jobs)
    D[j] - D[i] >= t[i][j] - M * (1 - sum(k in Vehicles) x[i][j][k]); // Delivery time for each job
  forall(i in Jobs, j in Jobs)
    T[i] >= D[i] - d[i]; // t if the delivery time of a customer exceeds its due date, then the tardiness for that customer equals the difference between the delivery time and the due date
 }
```

Figure 1 Model File for Experiment run 1

```
/*********************************************
 * OPL 22.1.1.0 Data
 * Author: bhave
 * Creation Date: 02-Dec-2023 at 10:19:16 PM
 *********************************************/

c = [[0 10 15],
     [10 0 35],
     [15 35 0]];

t = [[0 5 10],
     [5 0 20],
     [10 20 0]];

p = [25, 36];
w = [2, 3];
d = [12, 36];
F = 26;
Q = [2,2];
M = 1000;

```

Figure 2 Data file for Experiment run 1

6. Run simulation:

We created ten datasets to run the code. Each dataset includes a different number of vehicles, a different number of customers and jobs, and different fixed costs for the vehicles. Each data set includes a different number of processing times for each customer and different costs for processing the orders. The capacity of the vehicle is changed and altered to ensure that the above formulation works in different scenarios. A large number such as 14 customers and 1 origin is used which comes down to a 15x15 matrix to make sure that the formulation could handle a mediocre-sized dataset.

6.1 Experimental Run 1:

The dataset was considered for the first experimental run. It is a very simple network with one origin and two customers. The fixed cost for operating the vehicles was fixed and does not change throughout the different experiment runs. Table 1 consists of the basic details of the network. The Table 2 contains the cost matrix for the following two customer points and Table 3 contains the time matrix. Table 4 contains the processing time, deadline for each job and the penalty weight if the order doesn't reach the customer before the due date for each job. Table 5 contains the max capacity of each vehicle.

Table 1 Basic details of the network (Experimental run 1)

Number of Jobs	2
Number of Vehicles	2
Fixed Cost	26
Nodes	3

Table 4 contains the processing time, deadline for each job and the penalty weight if the order doesn't reach the customer before the due date for each job. Table 5 contains the max capacity of each vehicle

Table 2 Cost Matrix (Experimental run 1)

TRAVEL COST	ORIGIN	CUSTOMER 1	CUSTOMER 2
ORIGIN	0	10	15
CUSTOMER 1	10	0	35
CUSTOMER 2	15	35	0

These data were used to run the first experimental run. These belong to the first dataset among the 10 datasets that we created for this experimental run.

Table 3 Time Matrix (Experimental run 1)

TRAVEL TIME	ORIGIN	CUSTOMER 1	CUSTOMER 2
ORIGIN	0	5	10
CUSTOMER 1	5	0	20
CUSTOMER 2	10	20	0

Table 4 Job data (Experimental run 1)

DATA	JOB 1	JOB 2
PROCEESSING TIME	25	36
PENALTY WEIGHT	2	3
DUE DAYS	12	36

Table 5 Vehicle Capacity (Experimental run 1)

VEHICLE 1	VEHICLE 2
2	2

While running the code with the above details, we get the following results. Figure 1 represents the objective value of the function which is nothing but the total cost of the above network. The cost includes the sum of total travel cost of the vehicles, fixed cost of the vehicles used and the total penalty cost for the delivery tardiness.

```
// solution (optimal) with objective 102
// Quality Incumbent solution:
// MILP objective                                1.0200000000e+02
// MILP solution norm |x| (Total, Max)           2.20000e+01  1.00000e+01
// MILP solution error (Ax=b) (Total, Max)       0.00000e+00  0.00000e+00
// MILP x bound error (Total, Max)               0.00000e+00  0.00000e+00
// MILP x integrality error (Total, Max)         0.00000e+00  0.00000e+00
// MILP slack bound error (Total, Max)           9.09495e-13  4.54747e-13
//
```

Figure 3 Objective value (Experimental run 1)

The optimum value is comparatively low compared to the other experimental runs as there are no delivery tardiness. The other results are given in Figure 2 for this experimental run.

```
y = [1 1];
T = [0 0];
A = [[0 1]
              [0 0]];
S = [0 0];
D = [5 10];
```

Figure 4 Solution (Experimental run 1)

In Figure 2 there are 5 matrices that denotes different data that is essential for analysing the experimental run. The S matrix denotes the starting time of the vehicles for each job. Since there are two vehicles and two jobs, both the vehicles started at the same time. The D matrix denotes the delivery time of both the jobs. Here, since there are no delays, it took exactly the amount of time that was given in the set as travel time. The A matrix denotes which job took place first. Here in the first row, the second element has the value 1. This denotes that the second job was given priority than the first job. The T matrix denotes the delivery of each job. In this case, there are no delivery delays which is why it is denoted as 0 in both the cells. The y matrix denotes whether the vehicles were used or not. It holds the value of 1 if the vehicle is used and 0 otherwise.

6.2 Experimental Run 2

In this experimental run, we include 4 customers and 4 jobs. The fixed cost of the vehicle remains the same. However, the cost and time matrix for the following run is denoted in Table 6 and Table 7 respectively. The additional information regarding for all the four jobs are also mentioned in Table 8.

Table 6 Cost Matrix (Experimental run 2)

TRAVEL COST	ORIGIN	CUSTOMER 1	CUSTOMER 2	CUSTOMER 3	CUSTOMER 4
ORIGIN	0	7	51	28	19
CUSTOMER 1	7	0	15	36	31
CUSTOMER 2	51	15	0	27	18
CUSTOMER 3	28	36	27	0	25
CUSTOMER 4	19	31	18	25	0

The due dates have been updated and is denoted in Table 8. In this experiment run, four vehicles are used, and all the four vehicles have the same capacity as 2. The same data has been used in the upcoming experiment runs as well, to show the optimum value and the solution changes depending on the factors that corresponds to the total cost for this transportation problem.

Table 7 Time Matrix (Experimental run 2)

TRAVEL TIME	ORIGIN	CUSTOMER 1	CUSTOMER 2	CUSTOMER 3	CUSTOMER 4
ORIGIN	0	14	52	30	18
CUSTOMER 1	14	0	20	31	28
CUSTOMER 2	52	20	0	32	15
CUSTOMER 3	30	31	32	0	26
CUSTOMER 4	18	28	15	26	0

Table 8 Job Data (Experimental run 2)

DATA	JOB 1	JOB 2	JOB 3	JOB 4
PROCESSING TIME	10	20	30	23
PENALTY WEIGHT	2	3	4	5
DUE DAYS	60	60	60	60

Table 9 Vehicle Capacity (Experiment run 2)

VEHICLE 1	VEHICLE 2	VEHICLE 3	VEHICLE 4
2	2	2	2

With the above data, the objective value we get is 236 which is mentioned in Figure 3. The total cost has been increased from the previous experiment run, as the parameters have been altered and the number of jobs, vehicles and the job data has been increased.

```
// solution (optimal) with objective 236
// Quality Incumbent solution:
// MILP objective                                   2.3600000000e+02
// MILP solution norm |x| (Total, Max)              2.08000e+02  6.00000e+01
// MILP solution error (Ax=b) (Total, Max)          0.00000e+00  0.00000e+00
// MILP x bound error (Total, Max)                  0.00000e+00  0.00000e+00
// MILP x integrality error (Total, Max)            0.00000e+00  0.00000e+00
// MILP slack bound error (Total, Max)              0.00000e+00  0.00000e+00
//
```

Figure 5 Objective value (Experimental run 2)

Figure 4 shows us the data on how this run was allocated. Here as said earlier, The D matrix denotes the delivery time of all the four jobs. Again, there are no delays here as the due dates for the jobs were huge which gave the model to find a solution that does not involve extra tardiness costs. The T matrix denotes the delivery delay of each job. In this case, there are no delivery delays which is why it is denoted as 0 in all the four cells.

```
y = [0 1 1 1];
T = [0 0 0 0];
A = [[0 0 0 1]
             [1 0 1 1]
             [1 0 0 1]
             [0 0 0 0]];
S = [0 0 20 0];
D = [14 60 60 38];
```

Figure 6 Solution (Experimental run 2)

The y matrix denotes whether the vehicles were used or not. It holds the value of 1 if the vehicle is used and 0 otherwise. Here only three vehicles were used as there are only three elements in the y matrix which has the value 1.

6.3 Experimental run 3:

With the same data as in the experimental run 2, we are tweaking up the due dates for each of the jobs. This experiment run was done to showcase the effect of tardiness costs in this model which makes it unique. In the Table 10 it shows the due date for the corresponding jobs. The due dates for each of the jobs have been reduced drastically to showcase how it affects the optimum value.

Table 10 Job Data (Experimental run 3)

DATA	JOB 1	JOB 2	JOB 3	JOB 4
PROCEESSING TIME	10	20	30	23
PENALTY WEIGHT	2	3	4	5
DUE DAYS	2	1	2	1

From the Figure 5 it is clear that the optimum value which is the total cost for this transportation network has been increased up to 400. The effect of tardiness of the delivery for each job can be seen clearly in this experimental run.

```
// solution (optimal) with objective 665
// Quality Incumbent solution:
// MILP objective                                6.6500000000e+02
// MILP solution norm |x| (Total, Max)           3.02000e+02  5.40000e+01
// MILP solution error (Ax=b) (Total, Max)       0.00000e+00  0.00000e+00
// MILP x bound error (Total, Max)               0.00000e+00  0.00000e+00
// MILP x integrality error (Total, Max)         0.00000e+00  0.00000e+00
// MILP slack bound error (Total, Max)           0.00000e+00  0.00000e+00
//
```

Figure 7 Objective value (Experimental run 3)

In the Figure 6 the data for this experimental run is provided. The D matrix denotes the delivery time of all the four jobs. It can be seen that the delivery time took for each of the jobs has been changed from the previous run. From the T matrix which denotes the delivery delay for each of the job can be known that how long was the product delayed from its due date.

```
y = [0 1 1 1];
T = [32 53 28 17];
A = [[0 0 1 1]
             [1 0 1 1]
             [0 0 0 1]
             [0 0 0 0]];
S = [0 0 20 0];
D = [34 54 30 18];
```

Figure 8 Solution (Experimental run 3)

This shows us the impact of the due dates of each of the jobs. It is significant that the due dates should be breathable so that the tardiness penalty does not affect the total cost of the transportation problem.

6.4 Experimental run 4:

With the same data as in experimental run 2, now the number of vehicles has been reduced and the vehicle capacity has been increased drastically to monitor whether the vehicle capacity plays a role in reducing the overall cost. The Table 11 provides us the capacity of each vehicle that is used.

Table 11 Vehicle Capacity (Experiment run 4)

VEHICLE 1	VEHICLE 2
40	40

From the objective value from Figure 7 it is evident that the total cost for this transportation model has increased. Although the vehicle had increased capacity its deficiency in its total number affected the objective value.

```
// solution (optimal) with objective 862
// Quality Incumbent solution:
// MILP objective                                8.6200000000e+02
// MILP solution norm |x| (Total, Max)           4.86000e+02  8.30000e+01
// MILP solution error (Ax=b) (Total, Max)       0.00000e+00  0.00000e+00
// MILP x bound error (Total, Max)               0.00000e+00  0.00000e+00
// MILP x integrality error (Total, Max)         0.00000e+00  0.00000e+00
// MILP slack bound error (Total, Max)           0.00000e+00  0.00000e+00
//
```

Figure 9 Objective value (Experimental run 4)

And since there was only two vehicles to cover up all the four jobs the delivery delay for each of the jobs has shot up. We also need to consider that the due dates from the experimental run 3 have not been changed for this run. We still continue with the same due dates we used for the third run. From the S matrix we can deduce that the second vehicle has started first.

```
y = [1 1];
T = [81 62 28 47];
A = [[0 0 1 1]
             [1 0 1 1]
             [0 0 0 1]
             [0 0 0 0]];
S = [30 0];
D = [83 63 30 48];
```

Figure 10 Solution (Experimental run 4)

6.5 Experimental Run 5:

Now we increase the penalty weight for a job if fails to complete it before the due date while maintaining a tight deadline and with limited number of vehicles but with huge vehicle capacity as the previous experimental run where both the vehicles as 40 each. Considering the huge processing time and very low number of days given to deliver it is clear that the objective value or the total cost is going to rise enormously.

Table 12 Job Data (Experimental run 3)

DATA	JOB 1	JOB 2	JOB 3	JOB 4
PROCEESSING TIME	29	54	40	63
PENALTY WEIGHT	25	35	45	55
DUE DAYS	2	1	2	1

From the Figure 9 it is clear that the total cost has been increased drastically because of the sensitivity analysis that was done and the tweaking of all the parameters which contributes to the total cost of the transportation model.

```
// solution (optimal) with objective 14117
// Quality Incumbent solution:
// MILP objective                                 1.4117000000e+04
// MILP solution norm |x| (Total, Max)            8.98000e+02  1.28000e+02
// MILP solution error (Ax=b) (Total, Max)        0.00000e+00  0.00000e+00
// MILP x bound error (Total, Max)                0.00000e+00  0.00000e+00
// MILP x integrality error (Total, Max)          0.00000e+00  0.00000e+00
// MILP slack bound error (Total, Max)            0.00000e+00  0.00000e+00
//
```

Figure 11 Objective value (Experimental run 5)

The start time of the vehicle is huge because of the huge processing time for each job. And from the y matrix it is evident that both the vehicles have been used to complete this network.

```
y = [1 1];
T = [106 127 82 57];
A = [[0 0 0 0]
             [1 0 0 1]
             [1 1 0 1]
             [1 0 0 0]];
S = [40 94];
D = [108 128 84 58];
```

Figure 12 Solution (Experimental run 5)

6.6 Experimental Run 6:

In this experimental run, the size of the matrix is huge. This involves a network of fourteen customers and including the origin, which comes with a 15x15 matrix. This huge matrix took a long time to solve. It took almost nine hours and yet still it did not solve this problem. It is better to reduce the run time limit in the code to get a solution which is heuristic and not optimal. The snippet can be found in Figure 11 to limit the run time for the model.

```
execute {
  cplex.tilim = 300; // Setting time limit to 300 seconds
}
```

Figure 13 Snippet to limit the run time.

We limited the run time to 300 seconds to get a heuristic solution for this model. The solutions can be found in Figure 12 and Figure 13 for this network.

```
// solution (time limit exceeded) with objective 11282.9999998046
// Quality Incumbent solution:
// MILP objective                                  1.1283000000e+04
// MILP solution norm |x| (Total, Max)             1.41200e+03  9.00000e+01
// MILP solution error (Ax=b) (Total, Max)         9.51204e-11  9.09495e-13
// MILP x bound error (Total, Max)                 0.00000e+00  0.00000e+00
// MILP x integrality error (Total, Max)           0.00000e+00  0.00000e+00
// MILP slack bound error (Total, Max)             1.70048e-08  2.00089e-09
//
```

Figure 14 Objective Value (Experimental run 6)

```
y = [1 1 1 1 1 1 1 1 1 1];
T = [44 37 1 34 2 0 28 71 0 25 40 48 7 4];
A = [[0 0 1 0 1 1 1 0 0 1 0 1 0 1]
             [1 0 1 0 1 1 1 0 0 1 0 1 0 1]
             [0 0 0 0 1 0 0 0 0 0 0 0 0 1]
             [1 1 1 0 1 1 1 1 1 1 1 1 0 1]
             [0 0 0 0 0 0 0 0 0 0 0 0 0 1]
             [0 0 1 0 1 0 1 0 0 1 0 1 0 1]
             [0 0 1 0 1 0 0 0 0 0 0 1 0 1]
             [1 1 1 0 1 1 1 0 1 1 0 1 0 1]
             [1 1 1 0 1 1 1 0 0 1 0 1 0 1]
             [0 0 1 0 1 0 1 0 0 0 0 1 0 1]
             [1 1 1 0 1 1 1 1 1 1 0 1 0 1]
             [0 0 1 0 1 0 0 0 0 0 0 0 0 1]
             [1 1 1 1 1 1 1 1 1 1 1 1 0 1]
             [0 0 0 0 0 0 0 0 0 0 0 0 0 0]];
S = [18 44 30 0 0 0 63 0 0 6];
D = [77 52 48 56 41 50 56 90 41 39 77 60 63 35];
```

Figure 15 Solution (Experimental run 6)

All the above experimental runs were done to measure how each parameter and decision variables play a role in affecting the objective value of the model, which is in this case the total cost for the transportation including transportation cost, fixed cost and tardiness penalty cost.

7. Conclusion:

To sum up, the complex relationship that exists between supply chain management, logistics of transportation, and scheduling complexity highlights how important effective supply chain scheduling is to the success of businesses in the transportation sector. In order to optimize their operations and reduce overall costs, manufacturers must navigate a complex transportation network, which presents a number of challenges that have been examined in this study. This paper's distinctive contribution is its focus on the optimization problem's time-sensitive aspect. The study acknowledges the consequences that delays in the delivery process have in the real world by including the element of deadlines and the associated penalties for tardiness. The problem statement is further improved by adding fixed vehicle operating costs and travel expenses. This brings it closer to the well-known Vehicle Routing Problem (VRP), which is known for its intrinsic computational complexity.

8. Other related journals:

Additionally, two more papers on these topics were considered and was similar to the research paper that we analysed.

The first paper that we found similar to our major report was *An Exact Algorithm for the Asymmetrical Capacitated Vehicle Routing Problem* [2].

The vehicle routing problem with time windows and multiple-use vehicles is solved using a branch-and-price method in this article. This method creates a restricted master problem (RMP) by a column generation process, where each column represents a potential route or shift for the vehicles. Then, the RMP is repeatedly solved, with two factors regulating the generation of new, less expensive columns.

The pricing subproblems, which are essentially shortest path problems with additional restrictions (resource constraints), are solved by a label correcting algorithm. It considers the constraints of the issue, such as the maximum number of vehicles that can travel, the time windows that can be utilized, and the distances of the routes that can be taken.

The formulation's primary objective is to maximize revenue while minimizing the fleet's overall driving distance. This is accomplished by scheduling workdays that are appropriate for the cars and ensuring that every client is attended to within the allocated time. Additionally, this is accomplished by ensuring that the vehicle's capacity is not surpassed.

The optimal set of routes that serve every user at the lowest total cost—considering both the distance travelled and the money earned—are provided by the solution to the RMP problem. This algorithm performs particularly well during the workday when vehicles must travel multiple routes, such as when delivering perishable goods.

The second paper we found similar to our paper was *An integrated production and transportation scheduling problem with order acceptance and resource allocation decisions* [3].

To solve an integrated supply chain scheduling problem, the article presents a Mixed-Integer Programming (MIP) model in conjunction with two meta-heuristic algorithms: the Adaptive Genetic Algorithm (AGA) and Tabu Search (TS). The issue is classified as strong NP-Hard, meaning that solving it computationally will be challenging, particularly in large-scale scenarios.

The goal of the MIP model's objective function is to optimize the overall value of the supply chain system while taking into consideration the costs of transportation, additional resource allocation, penalties for delivery delays, and income from accepted jobs.

Statistical tests, including group comparisons and normality tests, supported the findings. According to the study, TS is superior at locating the optimal solution quickly, but AGA is more successful at locating the optimal solution overall. In addition to highlighting the value of well-coordinated decisions in supply chain management, the paper makes recommendations for future research on multi-objective optimization techniques to address the limitations of the current issue and a variety of real-world scenarios.

9. Teamwork:

Duties	Members Responsible
Coding	Bhavesh, Naeem
Dataset Preparation	Bhavesh, Naeem
Experimental Runs	Bhavesh, Naeem
Report Generation	Bhavesh, Naeem

10. References:

[1] Tamannaei, M., & Rasti-Barzoki, M. (2019). Mathematical programming and solution approaches for minimizing tardiness and transportation costs in the supply chain scheduling problem. *Computers & Industrial Engineering*, *127*, 643–656. https://doi.org/10.1016/j.cie.2018.11.003

[2] Laporte, G., Mercure, H., Nobert, Y., & du, U. (1984). *An Exact Algorithm for the Asymmetrical Capacitated Vehicle Routing Problem*. Montréal : Centre de recherche en gestion, Université du Québec à Montréal.

[3] Aminzadegan, S., Tamannaei, M., & Fazeli, M. (2021). An integrated production and transportation scheduling problem with order acceptance and resource allocation decisions. *Applied Soft Computing*, *112*, 107770. https://doi.org/10.1016/j.asoc.2021.107770