# ASSIGNMENT AND VEHICLE SCHEDULING PROBLEM USING REAL-TIME TRAFFIC DATA

**Narayan Kamdar**
**Nihar Jain**
**Parth Hindocha**

***Abstract: This paper reviews the applications of Operations Research in Supply Chain Management. This study examined how truck full load of goods be transported from point of origin to point of consumption within and outskirts of the Island City of India, Mumbai. Eight different supply centres were selected & with help of Hungarian method, each one was allocated eight different times. With the assistance of Google maps real-time traffic data has been collected in order to allocate the truck at a specific time to a specific location. Screenshots form a part of data analysis which proves to be a fruitful tool of the modern technology imbibed in smartphones. We assume that the factory has one truck and it has to deliver the supplies to 8 different locations in the 24-hour period. In addition to that the truck has to empty the supplies and return to the point of origin in the stipulated time. It is observed that it is not necessary that the traffic on different routes is constant which means that the amount of traffic on the route from the point of origin to a particular point of destination is not constant throughout the 24 hour period. Thus, through Hungarian method, different departure time slots were assigned to different destinations in order to achieve optimality.***

Assignment is a fundamental combinatorial problem in Operations Research. The objective of Assignment is to maximise profits or minimize costs by assigning the right people to the right jobs. Assignment and Optimization in businesses are of vital importance. It helps businesses to use the available resources in the best possible manner to cut down on costs and maximize their profits. In this research paper we have tried to achieve optimality through the tools of Operations Research with the data we acquired from Google Maps. The problem we have touched upon in this paper is known as Vehicle Scheduling Problem or Vehicle Assignment Problem.

The business we have considered in our study is a Distribution Centre (Fig. 1). A distribution centre has various costs: packaging, salaries, administration, warehousing. However, Transport forms a major portion of it. A distribution centre can only be successful if it makes best use of its distribution network. Thus, the problem we're looking to solve with our study is how the Distribution Centre can transport goods quickly, safely and efficiently with its limited resources, while making maximum profits. There basically are two solutions. The most direct solution is to build more and better-quality infrastructures such as roads, highways, rails etc. However, it is infeasible due to the high costs and other environmental and social constraints. The second solution to control and changes the internal/external factors that affect the channel of distribution like vehicles used, people employed, safety etc. What about the Traffic? Will choosing the right place at the right time help them in cutting down costs?

This Research Paper presents a scalable, simple solution for allocation of business resources in the best possible manner to gain optimal profits and minimise costs. We have used all the reliable and free information available across the internet. For example, we have used Google Maps to obtain the real time traffic data. This paper will have companies to make intelligent decisions for a safer faster and convenient trip.


Assignment problems arise in different situation where we have to find an optimal way to assign 'x' objects to 'y' other objects in an injective fashion. The assignment problems are a well-studied topic in combinatorial optimization. These problems find numerous applications in production planning, telecommunication VLSI design, economic etc. The assignment problem is a special case of Transportation problem (Shweta Singh, 2012)
Transportation sector, as the everyday carrier of millions of tons of freight and numbers of passengers, is the irreplaceable and irrefutable foundation of economic and industrial development. Even the advancements in building a virtual world, which hoped to cut down on physical transportation by virtual transference, not only could not decline the worldwide dependency on the sector, but also a rapid growth is being witnessed (Khodakaram Salimifard, 2012)
Traffic management is the planning, monitoring and control or influencing of traffic
It aims to:

- Maximise the effectiveness of the use of existing infrastructure;
- Ensure reliable and safe operation of transport

For road transport, tactical traffic management involves monitoring the actual traffic situation in real-time (including volumes, speeds, incidents, etc.) and then controlling or influencing the flow using the information in assistance with Global Positioning System(GPS) & Google Maps in order to reduce congestion, deal with incidents and improve network efficiency, safety and environmental performance, or achieve other objectives.

Roads account for most inland passenger and freight transport in the financial capital of the country, Mumbai. Road capacities in Mumbai have crossed the point of satiety, while interurban passenger and freight transport by road is increasing as a result of socio-economic and political changes that are increasing personal mobility and intra-city trade.

Road traffic management research at the urban level covers aspects such as intersection-based traffic management in urban areas, real-time traffic signal and management (Claus Seibt, 2009)

Cities and traffic have developed hand-in-hand since the earliest large human settlements. The same forces that draw inhabitants to converge in large urban areas also lead to sometimes intolerable levels of traffic congestion on urban streets. (Shekhar K Rahane, 2014)

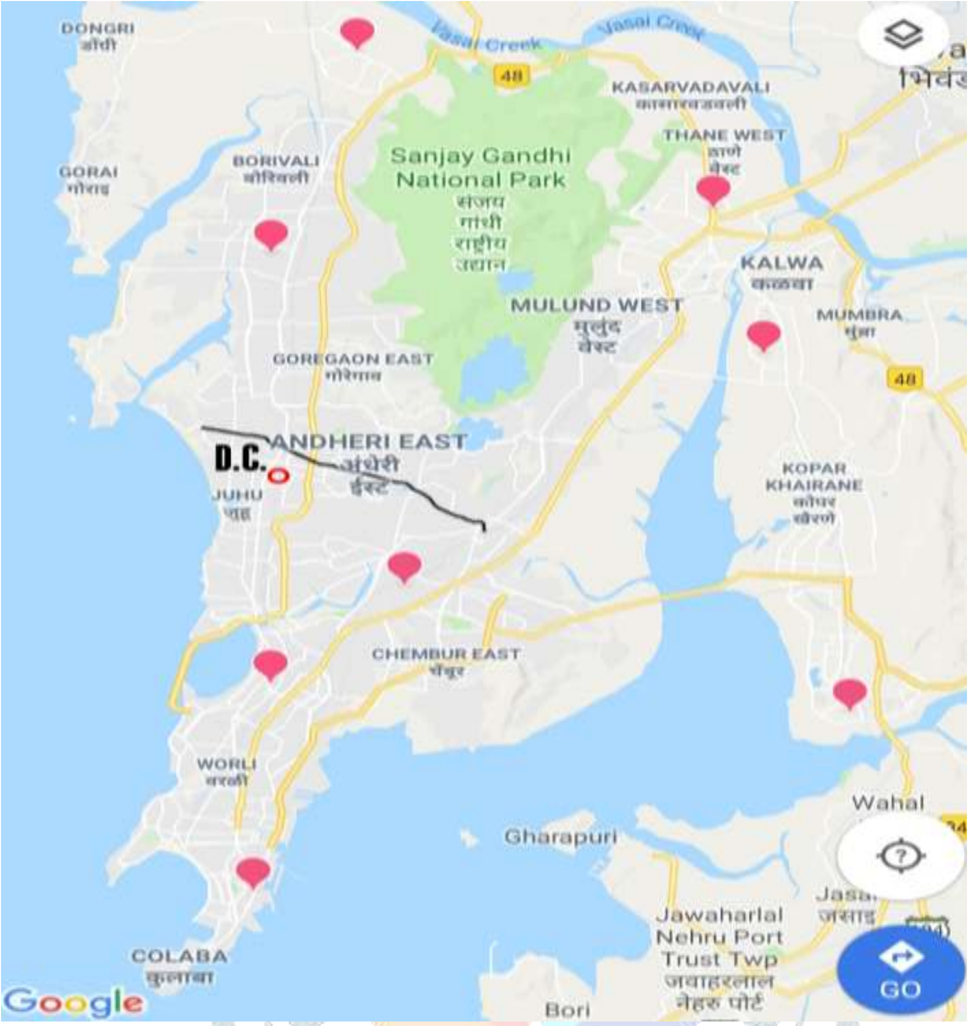
India’s transport sector is large and diverse; it caters to the transport needs of 1.1 billion people. In 2012-2013, the sector contributed about 5.2 per cent to the nation’s GDP, with road transportation having a major share of it. Good physical connectivity in urban and rural areas is essential for economic growth. An industry which is growing at such a faster pace and contributing a chunk of nation’s Gross Domestic Product (GDP), an efficient way of assigning the routes so as to achieve shorter duration of travel would boost the industry to leaps & bounds. Since the early 1990s, India's growing economy has witnessed a rise in demand for transport infrastructure and services. Efficient and reliable urban transport systems are crucial for India to sustain high economic growth. The significance of urban transport in India stems from the role that it plays in reduction of poverty, by improving access to labour markets and thus increasing incomes in poorer communities. Services and manufacturing industries particularly concentrate around major urban areas, and require efficient and reliable urban transport systems to move workers and connect production facilities to the logistics chain.

Mobility flows have become a key dynamic in the rapid urbanisation process of Indian cities with urban transport infrastructure constituting the skeleton of the urban form. Despite the increasing levels of urban mobility in Indian cities, access to places, activities and services is becoming increasingly difficult in terms of convenience, cost and time. In fact, present levels of urban mobility are already generating a crisis situation characterized by high levels of congestion, environmental pollution, traffic fatalities and inequity eventually leading to a situation of undesired accessibility crisis (Indian Institute for Human Settlements, 2014)

Methodology and Procedure:

The data has been collected from primary sources. Following are the sources we have used:
1. Google Maps.
2. Distribution Centre’s supply locations

The Distribution Centre in question, located in Vile Parle, has
1 locomotive,
8 destinations (Airoli, Chembur, Dadar, Fort, Kandivali, Mira Road, Seawoods and Thane) [Fig. 2] across the city,
8 time slots (12 midnight, 3AM, 6AM, 9AM, 12 noon, 3PM, 6PM, 9PM)




(FIG. 2)

For the purpose of this model we recorded the traffic and the time taken to reach from the Source (Distribution Centre) to the 8 Destinations over a period of 20 days. With the help of this data we tried to analyse the time of highest/lowest traffic and find the best route which the locomotive can take to reach there by using the least amount of resources.

[Fig. 2]

Assignment Problem:

The classical assignment problem was formulated in 1952 by D.F. Votaw and A. Orden [1] as a type of transportation problem. The mathematical model of the assignment problem for parallel processing can be described by an objective function minimizing the total cost of realizing processes:

$$\min z : z = \sum_{i=1}^{n} \sum_{j=1}^{n} k_{ij} \cdot x_{ij}, \qquad (1)$$

and the following constraints:

$$\sum_{i=1}^{n} x_{ij} = 1, \quad j = 1, 2, \ldots, n, \qquad (2)$$

$$\sum_{j=1}^{n} x_{ij} = 1, \quad i = 1, 2, \ldots, n, \qquad (3)$$

$$x_{ij} \in \{0, 1\}, \quad i = 1, 2, \ldots, n, \quad j = 1, 2, \ldots, n. \qquad (4)$$

*Where:*

$x_{ij}$– a binary variable for modelling a decision of selecting processes' modes; the variable assumes the value of 1 if the activity *i* is to be executed by the resource (crew) *j*, and equals 0 in the other case.

According to equations (2) and (3), considering equal numbers of crews and activities, each crew may be assigned only to one process and each process may be realized by only one crew.

H.W. Kuhn [2] in 1955 created the Hungarian method – exact algorithm for solving the model. He combined the ideas of two Hungarian mathematicians: D. König [3] and

J. Egerváry [4]. The method finds an optimal assignment for a given square cost matrix, and consist of five steps [2, 5–7]:
Step 1 – Subtract the smallest entry in each row from all the entries of its row.
Step 2 – Subtract the smallest entry in each column from all the entries of its column.
Step 3 – Construct a minimal number of lines, which covers all the zero entries of the cost matrix with $k_{ij}$.
Step 4 – If number of covering lines is $n$, then it is complete. Otherwise, proceed to
Step 5 – Determine the smallest, uncovered entry and subtract it from uncovered rows, and then add this entry to each covered column. Repeat Step 3.

(TOMCZAK, 2014)

We have made a few assumptions:

1. The locomotive completes one journey at a time. It takes a full truck load of goods, supplies it at the demand centre and returns to the factory.
2. We have observed, recorded and taken an average of the time required to reach from the source to the destination over a couple of weeks.

This is the Original Average Time Matrix:

| | **Dadar** | **Fort** | **Thane** | **Chembur** | **Kandivali** | **Mira Road** | **Seawoods** | **Airoli** |
|---|---|---|---|---|---|---|---|---|
| **12 PM** | 40.0 | 62.2 | 55.2 | 35.2 | 38.8 | 59.2 | 67.0 | 56.6 |
| **3 PM** | 36.4 | 57.0 | 50.0 | 30.6 | 40.2 | 54.4 | 72.2 | 48.6 |
| **6 PM** | 31.2 | 59.6 | 69.4 | 30.8 | 54.6 | 66.8 | 80.6 | 58.2 |
| **9 PM** | 33.0 | 52.2 | 62.4 | 33.0 | 49.6 | 64.2 | 78.6 | 51.6 |
| **12 AM** | 25.8 | 44.8 | 44.6 | 25.0 | 29.6 | 45.4 | 58.2 | 41.2 |
| **3 AM** | 22.0 | 34.4 | 36.0 | 24.8 | 28.6 | 34.0 | 48.4 | 29.4 |
| **6 AM** | 20.2 | 36.8 | 37.4 | 26.0 | 28.2 | 37.6 | 47.2 | 33.6 |
| **9 AM** | 31.0 | 52.0 | 56.2 | 30.4 | 41.8 | 43.8 | 68.6 | 49.8 |

**Subtract row minima**

We subtract the row minimum from each row:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 4.8 | 27.0 | 20.0 | 0.0 | 3.6 | 24.0 | 31.8 | 21.4 | (-35.2) |
| 5.8 | 26.4 | 19.4 | 0.0 | 9.6 | 23.8 | 41.6 | 18.0 | (-30.6) |
| 0.4 | 28.8 | 38.6 | 0.0 | 23.8 | 36.0 | 49.8 | 27.4 | (-30.8) |
| 0.0 | 19.2 | 29.4 | 0.0 | 16.6 | 31.2 | 45.6 | 18.6 | (-33) |
| 0.8 | 19.8 | 19.6 | 0.0 | 4.6 | 20.4 | 33.2 | 16.2 | (-25) |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 0.0 | 12.4 | 14.0 | 2.8 | 6.6 | 12.0 | 26.4 | 7.4 | (-22) |
| 0.0 | 16.6 | 17.2 | 5.8 | 8.0 | 17.4 | 27.0 | 13.4 | (-20.2) |
| 0.6 | 21.6 | 25.8 | 0.0 | 11.4 | 13.4 | 38.2 | 19.4 | (-30.4) |

**Subtract column minima**
We subtract the column minimum from each column:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 4.8 | 14.6 | 6.0 | 0.0 | 0.0 | 12.0 | 5.4 | 14.0 |
| 5.8 | 14.0 | 5.4 | 0.0 | 6.0 | 11.8 | 15.2 | 10.6 |
| 0.4 | 16.4 | 24.6 | 0.0 | 20.2 | 24.0 | 23.4 | 20.0 |
| 0.0 | 6.8 | 15.4 | 0.0 | 13.0 | 19.2 | 19.2 | 11.2 |
| 0.8 | 7.4 | 5.6 | 0.0 | 1.0 | 8.4 | 6.8 | 8.8 |
| 0.0 | 0.0 | 0.0 | 2.8 | 3.0 | 0.0 | 0.0 | 0.0 |
| 0.0 | 4.2 | 3.2 | 5.8 | 4.4 | 5.4 | 0.6 | 6.0 |
| 0.6 | 9.2 | 11.8 | 0.0 | 7.8 | 1.4 | 11.8 | 12.0 |
| | (-12.4) | (-14) | | (-3.6) | (-12) | (-26.4) | (-7.4) |

**Cover all zeros with a minimum number of lines**
There are 4 lines required to cover all zeros:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **4.8** | **14.6** | **6.0** | **0.0** | **0.0** | **12.0** | **5.4** | **14.0** | **x** |
| **5.8** | 14.0 | 5.4 | **0.0** | 6.0 | 11.8 | 15.2 | 10.6 | |
| **0.4** | 16.4 | 24.6 | **0.0** | 20.2 | 24.0 | 23.4 | 20.0 | |
| **0.0** | 6.8 | 15.4 | **0.0** | 13.0 | 19.2 | 19.2 | 11.2 | |
| **0.8** | 7.4 | 5.6 | **0.0** | 1.0 | 8.4 | 6.8 | 8.8 | |
| **0.0** | **0.0** | **0.0** | **2.8** | **3.0** | **0.0** | **0.0** | **0.0** | **x** |
| **0.0** | 4.2 | 3.2 | **5.8** | 4.4 | 5.4 | 0.6 | 6.0 | |
| **0.6** | 9.2 | 11.8 | **0.0** | 7.8 | 1.4 | 11.8 | 12.0 | |
| **x** | | | **x** | | | | | |

**Create additional zeros**

The number of lines is smaller than 8. The smallest uncovered number is 0.6. We subtract this number from all uncovered elements and add it to all elements that are covered twice:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 5.4 | 14.6 | 6.0 | 0.6 | 0.0 | 12.0 | 5.4 | 14.0 |
| 5.8 | 13.4 | 4.8 | 0.0 | 5.4 | 11.2 | 14.6 | 10.0 |
| 0.4 | 15.8 | 24.0 | 0.0 | 19.6 | 23.4 | 22.8 | 19.4 |
| 0.0 | 6.2 | 14.8 | 0.0 | 12.4 | 18.6 | 18.6 | 10.6 |
| 0.8 | 6.8 | 5.0 | 0.0 | 0.4 | 7.8 | 6.2 | 8.2 |
| 0.6 | 0.0 | 0.0 | 3.4 | 3.0 | 0.0 | 0.0 | 0.0 |
| 0.0 | 3.6 | 2.6 | 5.8 | 3.8 | 4.8 | 0.0 | 5.4 |
| 0.6 | 8.6 | 11.2 | 0.0 | 7.2 | 0.8 | 11.2 | 11.4 |

**Cover all zeros with a minimum number of lines**

There are 5 lines required to cover all zeros:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **5.4** | **14.6** | **6.0** | **0.6** | **0.0** | **12.0** | **5.4** | **14.0** | **x** |
| 5.8 | 13.4 | 4.8 | **0.0** | 5.4 | 11.2 | 14.6 | 10.0 | |
| 0.4 | 15.8 | 24.0 | **0.0** | 19.6 | 23.4 | 22.8 | 19.4 | |
| **0.0** | **6.2** | **14.8** | **0.0** | **12.4** | **18.6** | **18.6** | **10.6** | **x** |
| 0.8 | 6.8 | 5.0 | **0.0** | 0.4 | 7.8 | 6.2 | 8.2 | |
| **0.6** | **0.0** | **0.0** | **3.4** | **3.0** | **0.0** | **0.0** | **0.0** | **x** |
| **0.0** | **3.6** | **2.6** | **5.8** | **3.8** | **4.8** | **0.0** | **5.4** | **x** |
| 0.6 | 8.6 | 11.2 | **0.0** | 7.2 | 0.8 | 11.2 | 11.4 | |
| | | | x | | | | | |

**Create additional zeros**

The number of lines is smaller than 8. The smallest uncovered number is 0.4. We subtract this number from all uncovered elements and add it to all elements that are covered twice:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 5.4 | 14.6 | 6.0 | 1.0 | 0.0 | 12.0 | 5.4 | 14.0 |
| 5.4 | 13.0 | 4.4 | 0.0 | 5.0 | 10.8 | 14.2 | 9.6 |
| 0.0 | 15.4 | 23.6 | 0.0 | 19.2 | 23.0 | 22.4 | 19.0 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0.0 | 6.2 | 14.8 | 0.4 | 12.4 | 18.6 | 18.6 | 10.6 |
| 0.4 | 6.4 | 4.6 | 0.0 | 0.0 | 7.4 | 5.8 | 7.8 |
| 0.6 | 0.0 | 0.0 | 3.8 | 3.0 | 0.0 | 0.0 | 0.0 |
| 0.0 | 3.6 | 2.6 | 6.2 | 3.8 | 4.8 | 0.0 | 5.4 |
| 0.2 | 8.2 | 10.8 | 0.0 | 6.8 | 0.4 | 10.8 | 11.0 |

**Cover all zeros with a minimum number of lines**

There are 5 lines required to cover all zeros:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **5.4** | **14.6** | **6.0** | **1.0** | **0.0** | **12.0** | **5.4** | **14.0** | **x** |
| **5.4** | 13.0 | 4.4 | **0.0** | 5.0 | 10.8 | 14.2 | 9.6 | |
| **0.0** | 15.4 | 23.6 | **0.0** | 19.2 | 23.0 | 22.4 | 19.0 | |
| **0.0** | 6.2 | 14.8 | **0.4** | 12.4 | 18.6 | 18.6 | 10.6 | |
| **0.4** | 6.4 | 4.6 | **0.0** | 0.0 | 7.4 | 5.8 | 7.8 | |
| **0.6** | **0.0** | **0.0** | **3.8** | **3.0** | **0.0** | **0.0** | **0.0** | **x** |
| **0.0** | **3.6** | **2.6** | **6.2** | **3.8** | **4.8** | **0.0** | **5.4** | **x** |
| **0.2** | 8.2 | 10.8 | **0.0** | 6.8 | 0.4 | 10.8 | 11.0 | |
| **x** | | | **x** | | | | | |

**Create additional zeros**

The number of lines is smaller than 8. The smallest uncovered number is 3.5527136788005E-15. We subtract this number from all uncovered elements and add it to all elements that are covered twice:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 5.4 | 14.6 | 6.0 | 1.0 | 0.0 | 12.0 | 5.4 | 14.0 |
| 5.4 | 13.0 | 4.4 | 0.0 | 5.0 | 10.8 | 14.2 | 9.6 |
| 0.0 | 15.4 | 23.6 | 0.0 | 19.2 | 23.0 | 22.4 | 19.0 |
| 0.0 | 6.2 | 14.8 | 0.4 | 12.4 | 18.6 | 18.6 | 10.6 |
| 0.4 | 6.4 | 4.6 | 0.0 | 0.0 | 7.4 | 5.8 | 7.8 |
| 0.6 | 0.0 | 0.0 | 3.8 | 3.0 | 0.0 | 0.0 | 0.0 |
| 0.0 | 3.6 | 2.6 | 6.2 | 3.8 | 4.8 | 0.0 | 5.4 |
| 0.2 | 8.2 | 10.8 | 0.0 | 6.8 | 0.4 | 10.8 | 11.0 |

**Cover all zeros with a minimum number of lines**

There are 5 lines required to cover all zeros:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **5.4** | 14.6 | 6.0 | **1.0** | **0.0** | 12.0 | 5.4 | 14.0 | |
| **5.4** | 13.0 | 4.4 | **0.0** | **5.0** | 10.8 | 14.2 | 9.6 | |
| **0.0** | 15.4 | 23.6 | **0.0** | **19.2** | 23.0 | 22.4 | 19.0 | |
| **0.0** | 6.2 | 14.8 | **0.4** | **12.4** | 18.6 | 18.6 | 10.6 | |
| **0.4** | 6.4 | 4.6 | **0.0** | **0.0** | 7.4 | 5.8 | 7.8 | |
| **0.6** | **0.0** | **0.0** | **3.8** | **3.0** | **0.0** | **0.0** | **0.0** | **x** |
| **0.0** | **3.6** | **2.6** | **6.2** | **3.8** | **4.8** | **0.0** | **5.4** | **x** |
| **0.2** | 8.2 | 10.8 | **0.0** | **6.8** | 0.4 | 10.8 | 11.0 | |
| **x** | | | **x** | **x** | | | | |

**Create additional zeros**

The number of lines is smaller than 8. The smallest uncovered number is 0.39999999999999. We subtract this number from all uncovered elements and add it to all elements that are covered twice:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 5.4 | 14.2 | 5.6 | 1.0 | 0.0 | 11.6 | 5.0 | 13.6 |
| 5.4 | 12.6 | 4.0 | 0.0 | 5.0 | 10.4 | 13.8 | 9.2 |
| 0.0 | 15.0 | 23.2 | 0.0 | 19.2 | 22.6 | 22.0 | 18.6 |
| 0.0 | 5.8 | 14.4 | 0.4 | 12.4 | 18.2 | 18.2 | 10.2 |
| 0.4 | 6.0 | 4.2 | 0.0 | 0.0 | 7.0 | 5.4 | 7.4 |
| 1.0 | 0.0 | 0.0 | 4.2 | 3.4 | 0.0 | 0.0 | 0.0 |
| 0.4 | 3.6 | 2.6 | 6.6 | 4.2 | 4.8 | 0.0 | 5.4 |
| 0.2 | 7.8 | 10.4 | 0.0 | 6.8 | 0.0 | 10.4 | 10.6 |

**Cover all zeros with a minimum number of lines**

There are 6 lines required to cover all zeros:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **5.4** | 14.2 | 5.6 | **1.0** | **0.0** | 11.6 | 5.0 | 13.6 |
| **5.4** | 12.6 | 4.0 | **0.0** | **5.0** | 10.4 | 13.8 | 9.2 |
| **0.0** | 15.0 | 23.2 | **0.0** | **19.2** | 22.6 | 22.0 | 18.6 |
| **0.0** | 5.8 | 14.4 | **0.4** | **12.4** | 18.2 | 18.2 | 10.2 |
| **0.4** | 6.0 | 4.2 | **0.0** | **0.0** | 7.0 | 5.4 | 7.4 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **1.0** | **0.0** | **0.0** | **4.2** | **3.4** | **0.0** | **0.0** | **0.0** | **x** |
| **0.4** | **3.6** | **2.6** | **6.6** | **4.2** | **4.8** | **0.0** | **5.4** | **x** |
| **0.2** | **7.8** | **10.4** | **0.0** | **6.8** | **0.0** | **10.4** | **10.6** | **x** |
| **x** | | | **x** | **x** | | | | |

**Create additional zeros**
The number of lines is smaller than 8. The smallest uncovered number is 4. We subtract this number from all uncovered elements and add it to all elements that are covered twice:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 5.4 | 10.2 | 1.6 | 1.0 | 0.0 | 7.6 | 1.0 | 9.6 |
| 5.4 | 8.6 | 0.0 | 0.0 | 5.0 | 6.4 | 9.8 | 5.2 |
| 0.0 | 11.0 | 19.2 | 0.0 | 19.2 | 18.6 | 18.0 | 14.6 |
| 0.0 | 1.8 | 10.4 | 0.4 | 12.4 | 14.2 | 14.2 | 6.2 |
| 0.4 | 2.0 | 0.2 | 0.0 | 0.0 | 3.0 | 1.4 | 3.4 |
| 5.0 | 0.0 | 0.0 | 8.2 | 7.4 | 0.0 | 0.0 | 0.0 |
| 4.4 | 3.6 | 2.6 | 10.6 | 8.2 | 4.8 | 0.0 | 5.4 |
| 4.2 | 7.8 | 10.4 | 4.0 | 10.8 | 0.0 | 10.4 | 10.6 |

**Cover all zeros with a minimum number of lines**
There are 7 lines required to cover all zeros:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **5.4** | 10.2 | 1.6 | **1.0** | **0.0** | 7.6 | 1.0 | 9.6 | |
| **5.4** | **8.6** | **0.0** | **0.0** | **5.0** | **6.4** | **9.8** | **5.2** | **x** |
| **0.0** | 11.0 | 19.2 | **0.0** | **19.2** | 18.6 | 18.0 | 14.6 | |
| **0.0** | 1.8 | 10.4 | **0.4** | **12.4** | 14.2 | 14.2 | 6.2 | |
| **0.4** | 2.0 | 0.2 | **0.0** | **0.0** | 3.0 | 1.4 | 3.4 | |
| **5.0** | **0.0** | **0.0** | **8.2** | **7.4** | **0.0** | **0.0** | **0.0** | **x** |
| **4.4** | **3.6** | **2.6** | **10.6** | **8.2** | **4.8** | **0.0** | **5.4** | **x** |
| **4.2** | **7.8** | **10.4** | **4.0** | **10.8** | **0.0** | **10.4** | **10.6** | **x** |
| **x** | | | **x** | **x** | | | | |

**Create additional zeros**
The number of lines is smaller than 8. The smallest uncovered number is 0.2. We subtract this number from all uncovered elements and add it to all elements that are covered twice:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 5.4 | 10.0 | 1.4 | 1.0 | 0.0 | 7.4 | 0.8 | 9.4 |
| 5.6 | 8.6 | 0.0 | 0.2 | 5.2 | 6.4 | 9.8 | 5.2 |
| 0.0 | 10.8 | 19.0 | 0.0 | 19.2 | 18.4 | 17.8 | 14.4 |
| 0.0 | 1.6 | 10.2 | 0.4 | 12.4 | 14.0 | 14.0 | 6.0 |
| 0.4 | 1.8 | 0.0 | 0.0 | 0.0 | 2.8 | 1.2 | 3.2 |
| 5.2 | 0.0 | 0.0 | 8.4 | 7.6 | 0.0 | 0.0 | 0.0 |
| 4.6 | 3.6 | 2.6 | 10.8 | 8.4 | 4.8 | 0.0 | 5.4 |
| 4.4 | 7.8 | 10.4 | 4.2 | 11.0 | 0.0 | 10.4 | 10.6 |

**Cover all zeros with a minimum number of lines**
There are 7 lines required to cover all zeros:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **5.4** | 10.0 | **1.4** | **1.0** | **0.0** | 7.4 | 0.8 | 9.4 | |
| **5.6** | 8.6 | **0.0** | **0.2** | **5.2** | 6.4 | 9.8 | 5.2 | |
| **0.0** | 10.8 | **19.0** | **0.0** | **19.2** | 18.4 | 17.8 | 14.4 | |
| **0.0** | 1.6 | **10.2** | **0.4** | **12.4** | 14.0 | 14.0 | 6.0 | |
| **0.4** | 1.8 | **0.0** | **0.0** | **0.0** | 2.8 | 1.2 | 3.2 | |
| **5.2** | **0.0** | **0.0** | **8.4** | **7.6** | **0.0** | **0.0** | **0.0** | **x** |
| **4.6** | **3.6** | **2.6** | **10.8** | **8.4** | **4.8** | **0.0** | **5.4** | **x** |
| **4.4** | **7.8** | **10.4** | **4.2** | **11.0** | **0.0** | **10.4** | **10.6** | **x** |
| **x** | | **x** | **x** | **x** | | | | |

**Create additional zeros**
The number of lines is smaller than 8. The smallest uncovered number is 0.8. We subtract this number from all uncovered elements and add it to all elements that are covered twice:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 5.4 | 9.2 | 1.4 | 1.0 | 0.0 | 6.6 | 0.0 | 8.6 |
| 5.6 | 7.8 | 0.0 | 0.2 | 5.2 | 5.6 | 9.0 | 4.4 |
| 0.0 | 10.0 | 19.0 | 0.0 | 19.2 | 17.6 | 17.0 | 13.6 |
| 0.0 | 0.8 | 10.2 | 0.4 | 12.4 | 13.2 | 13.2 | 5.2 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0.4 | 1.0 | 0.0 | 0.0 | 0.0 | 2.0 | 0.4 | 2.4 |
| 6.0 | 0.0 | 0.8 | 9.2 | 8.4 | 0.0 | 0.0 | 0.0 |
| 5.4 | 3.6 | 3.4 | 11.6 | 9.2 | 4.8 | 0.0 | 5.4 |
| 5.2 | 7.8 | 11.2 | 5.0 | 11.8 | 0.0 | 10.4 | 10.6 |

**Cover all zeros with a minimum number of lines**

There are 7 lines required to cover all zeros:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **5.4** | 9.2 | **1.4** | **1.0** | **0.0** | 6.6 | **0.0** | 8.6 | |
| **5.6** | 7.8 | **0.0** | **0.2** | **5.2** | 5.6 | **9.0** | 4.4 | |
| **0.0** | 10.0 | **19.0** | **0.0** | **19.2** | 17.6 | **17.0** | 13.6 | |
| **0.0** | 0.8 | **10.2** | **0.4** | **12.4** | 13.2 | **13.2** | 5.2 | |
| **0.4** | 1.0 | **0.0** | **0.0** | **0.0** | 2.0 | **0.4** | 2.4 | |
| **6.0** | **0.0** | **0.8** | **9.2** | **8.4** | **0.0** | **0.0** | **0.0** | **x** |
| **5.4** | 3.6 | **3.4** | **11.6** | **9.2** | 4.8 | **0.0** | 5.4 | |
| **5.2** | **7.8** | **11.2** | **5.0** | **11.8** | **0.0** | **10.4** | **10.6** | **x** |
| **x** | | **x** | **x** | **x** | | **x** | | |

**Create additional zeros**

The number of lines is smaller than 8. The smallest uncovered number is 0.8. We subtract this number from all uncovered elements and add it to all elements that are covered twice:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 5.4 | 8.4 | 1.4 | 1.0 | 0.0 | 5.8 | 0.0 | 7.8 |
| 5.6 | 7.0 | 0.0 | 0.2 | 5.2 | 4.8 | 9.0 | 3.6 |
| 0.0 | 9.2 | 19.0 | 0.0 | 19.2 | 16.8 | 17.0 | 12.8 |
| 0.0 | 0.0 | 10.2 | 0.4 | 12.4 | 12.4 | 13.2 | 4.4 |
| 0.4 | 0.2 | 0.0 | 0.0 | 0.0 | 1.2 | 0.4 | 1.6 |
| 6.8 | 0.0 | 1.6 | 10.0 | 9.2 | 0.0 | 0.8 | 0.0 |
| 5.4 | 2.8 | 3.4 | 11.6 | 9.2 | 4.0 | 0.0 | 4.6 |
| 6.0 | 7.8 | 12.0 | 5.8 | 12.6 | 0.0 | 11.2 | 10.6 |

**Cover all zeros with a minimum number of lines**

There are 8 lines required to cover all zeros:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **5.4** | **8.4** | **1.4** | **1.0** | **0.0** | **5.8** | **0.0** | **7.8** | **x** |
| **5.6** | **7.0** | **0.0** | **0.2** | **5.2** | **4.8** | **9.0** | **3.6** | **x** |
| **0.0** | **9.2** | **19.0** | **0.0** | **19.2** | **16.8** | **17.0** | **12.8** | **x** |
| **0.0** | **0.0** | **10.2** | **0.4** | **12.4** | **12.4** | **13.2** | **4.4** | **x** |
| **0.4** | **0.2** | **0.0** | **0.0** | **0.0** | **1.2** | **0.4** | **1.6** | **x** |
| **6.8** | **0.0** | **1.6** | **10.0** | **9.2** | **0.0** | **0.8** | **0.0** | **x** |
| **5.4** | **2.8** | **3.4** | **11.6** | **9.2** | **4.0** | **0.0** | **4.6** | **x** |
| **6.0** | **7.8** | **12.0** | **5.8** | **12.6** | **0.0** | **11.2** | **10.6** | **x** |

**The optimal assignment**

Because there are 8 lines required, the zeros cover an optimal assignment:

| | Dadar | Fort | Thane | Chembur | Kandivali | Mira Road | Seawoods | Airoli |
|---|---|---|---|---|---|---|---|---|
| **12 PM** | 5.4 | 8.4 | 1.4 | 1.0 | **0.0** | 5.8 | 0.0 | 7.8 |
| **3 PM** | 5.6 | 7.0 | **0.0** | 0.2 | 5.2 | 4.8 | 9.0 | 3.6 |
| **6 PM** | **0.0** | 9.2 | 19.0 | 0.0 | 19.2 | 16.8 | 17.0 | 12.8 |
| **9 PM** | 0.0 | **0.0** | 10.2 | 0.4 | 12.4 | 12.4 | 13.2 | 4.4 |
| **12 AM** | 0.4 | 0.2 | 0.0 | **0.0** | 0.0 | 1.2 | 0.4 | 1.6 |
| **3 AM** | 6.8 | 0.0 | 1.6 | 10.0 | 9.2 | 0.0 | 0.8 | **0.0** |
| **6 AM** | 5.4 | 2.8 | 3.4 | 11.6 | 9.2 | 4.0 | **0.0** | 4.6 |
| **9 AM** | 6.0 | 7.8 | 12.0 | 5.8 | 12.6 | **0.0** | 11.2 | 10.6 |

This corresponds to the following optimal assignment in the original cost matrix:

| | Dadar | Fort | Thane | Chembur | Kandivali | Mira Road | Seawoods | Airoli |
|---|---|---|---|---|---|---|---|---|
| **12 PM** | 40.0 | 62.2 | 55.2 | 35.2 | **38.8** | 59.2 | 67.0 | 56.6 |
| **3 PM** | 36.4 | 57.0 | **50.0** | 30.6 | 40.2 | 54.4 | 72.2 | 48.6 |
| 6 PM | **31.2** | 59.6 | 69.4 | 30.8 | 54.6 | 66.8 | 80.6 | 58.2 |
| **9 PM** | 33.0 | **52.2** | 62.4 | 33.0 | 49.6 | 64.2 | 78.6 | 51.6 |
| **12 AM** | 25.8 | 44.8 | 44.6 | **25.0** | 29.6 | 45.4 | 58.2 | 41.2 |
| **3 AM** | 22.0 | 34.4 | 36.0 | 24.8 | 28.6 | 34.0 | 48.4 | **29.4** |
| **6 AM** | 20.2 | 36.8 | 37.4 | 26.0 | 28.2 | 37.6 | **47.2** | 33.6 |
| **9 AM** | 31.0 | 52.0 | 56.2 | 30.4 | 41.8 | **43.8** | 68.6 | 49.8 |

The optimal value equals 317.6.

| | |
|---|---|
| Dadar | 6 PM |
| Fort | 9 PM |
| Thane | 3 PM |
| Chembur | 12 AM |
| Kandivali | 12 PM |
| Mira Road | 9 AM |
| Seawoods | 6 AM |
| Airoli | 3 AM |

Data Analysis

Over the course of these few weeks we have noticed a general pattern in the traffic at different time intervals. The relative percentages of the same are given below. From this table we have tried to generalise and quantify the amount of traffic faced in order to reach the destination at different times. According to this data, 12 MIDNIGHT to 6PM the traffic on the streets is the least. It would be advisable to visit the centres which far during this time. And the traffic from 6PM to 9PM, is at its peak. At this time, if the good are transported to the nearest destination the cost and time incurred would be considerably low.

In conclusion, the results of this study provide us with information which is vital in the Supply Chain Management for industries. Operations Research and Supply Chain Management go hand in hand in today's fastmoving tech-driven world. In a world where a click is the solution to all problems, the necessity of addition of technology even in transporting goods reaps mobility and helps businesses grow at a faster pace. This paper focuses on cutting down transport costs and time by locomotive scheduling mathematical model. We have made sure to keep the solution simple and universal at the same time. We have made use of the technology that is easily available with everyone. These small tweaks in the transport system can cave thousands of rupees and several hours per day. The model helps in making decisions based on the following:

1. Time taken to reach from the source to the destination.
2. Locomotive scheduling.

Moving forward, other factors like time distance and transporting in 2 or more places at one time can be integrated in this model. With this paper we have managed to assign the locomotive and find the best/least time required to transport the goods from source to the destination. The results of this model are shown above.

References

(Claus Seibt, 2009) Traffic Management for land Transport
(Indian Institute for Human Settlements, 2014) Urban Transport in India Challenges and Recommendations
(Khodakaram Salimifard, 2012) Green Transportation and the role of Operation Research
(Shekhar K Rahane, 2014) Traffic Congestion – Causes and Solutions A study of Talegaon Dabhade City
(Shweta Singh, 2012) A Comparative Analysis of Assignment Problem
(TOMCZAK, 2014) Assignment Problem and its extensions for construction project scheduling