# Competitive analysis of linear programming and heuristic method in optimization of supply chain network

Aastha Singh
Electrical and Electronics Engineering
IIT Guwahati
Guwahati India
s.aastha@iitg.ac.in

Satyam Bansal
Electrical and Electronics Engineering
IIT Guwahati
Guwahati India
b.satyam@iitg.ac.in

***Abstract*— Globalization and specialisation are currently attracting industries from all over the world. Working with dependable strategic partners is the greatest way to address any problems brought on by competition. It is crucial that products are carried effectively and efficiently in the competitive economy of today, from raw material locations to processing plants, component manufacturers, final goods assembly plants, distribution hubs, merchants, and customers. A set of customer locations with demand and a set of potential facility locations are known ahead of time in this investigation. Having a facility at a potential site has a known fixed location cost. With regard to delivery lead times from each plant to the demand, the unit shipment costs for each eligible location are known to one another.. Under the premise that all customer expectations are met, lowering the overall cost, locating the facilities and the movement of shipments between such facilities and their clients is a difficulty. In the aforementioned study, we compare the effectiveness of a heuristic approach and a linear programming strategy.**

## 2. Introduction

A supply chain network is essentially a collection of strategies that uses information about suppliers, requests, production capacity, manufacturers, the proper distribution of manufacturing facility locations, etc., all with the goal of satisfying all demands. Each supply chain network might be different from the others depending on the locations of the production, the facilities available, and the types of requests.

However, they do share a characteristic. The final stage of the chain supply network involves material receiving and product production to meet demand.

The goal of all production facilities is to lower the overall cost of the two processes.

In supply chain management, determining actor positions and counts, the quantity of product movement between, and lowering transportation costs are all dealt with as network design problems turning it into an NP-hard the problem.

The aforementioned issue can be resolved utilising linear programming techniques like the simplex approach, according to numerous studies and researches. Although these techniques are renowned for offering the best answers, one disadvantage is that they require exponentially more time to produce an ideal outcome.

On the other hand, heuristic methods such as PSO, ACO, and greedy algorithms address the issue more quickly but produce results that are close to the best possible ones.

Considering the above factors this research compares the effectiveness of heuristic and linear programming techniques for supply chain network optimization.

We chose the Greedy algorithm since the other two heuristic approaches either employed only discrete parameters or made it difficult to integrate constraints in the other two algorithms, while a supply chain network mixes discrete and continuous factors.

## 3. MATHEMATICAL FORMULATION

This section gives a mathematical model for a distribution network that takes into account factors and constraints inspired from the real world.

### 3.1 Objectives of the model

The objective of the proposed study is to identify the best sites for manufacturing facilities so that all customer demands can be met within a given time frame while lowering production costs.

The final selections regarding the locations of each plant are made based on the plant's capabilities, market demand, variable production costs, transportation expenses each exported unit, and fixed costs every month at each plant, depending on whether it is a low- or high-capacity factory. To enhance our model, we added a second scenario that took into account the product's delivery lead time.
This work considers a constraint objective formulation of a supply chain network and analyses the use of simplex method to a heuristic approach using greedy algorithm.

### 3.2 Model Assumptions

The information given to us :

- Number of demand locations and their demanding capacity
- Number of potential plants and their production capacities
- Delivery lead time from a plant to a demand centre

### 3.3 Problem Description

The proposed research employs heuristics and linear programming to determine the best location for the plant and whether it should be running at full or reduced capacity.
Numerous elements, some of which are listed below, were taken into account to reduce production costs.

- Manufacturing facility fixed costs which includes fixed cost of machines used, administrative fees,space rental etc.
- Variable costs which includes raw materials , freight costs etc.
- Manufacturing capacity on basis of weather the plant was operating on high capacity otr low capacity.
- Customers demand per market.
- Delivery lead time from manufacturing facilities to demand centres.

The used supply chain network requires the following parameters, sets and decision variables.

### 3.1 Indices and sets

i $\in I$ , represents plant locations
$j \in J$ , represents demand locations
$s \in S$ , represents type of manufacturing facility

### 3.2 Parameters

| Notations used | Meanings |
|---|---|
| m | No. of production facilities |
| n | Number of markets or demand centres |
| Dj | Annual demand for market j |
| Ki | Potential capacity of plant i |
| $f_{is}$ | Annualised fixed cost of keeping factory i size s open |
| $C_{ij}$ | Cost of production and shipping one unit from factory i to market j |
| Tij | Time taken to ship products from factory i to market j |
| t | Storage time of the product |

**Table 1**

### 3.3 Decision variables

| Variable used | Meaning |
|---|---|
| $y_{is}$ | An binary operator Takes value 1 if plant i of size s is open 0 otherwise |
| $x_{is}$ | Cost of producing and shipping one unit from factory i to market j |

**Table 2**

### 3.4 Constraints used

- Demand at each market should be satisfied .
  Constraint 1
  $\sum_{i=1}^{n} x_{ij} = D_{ij}$ , for j ∈ 1,2,3….m
- No plant can supply more than its capacity respective of the plant capacity chosen.
  Constraint 2
  $\sum_{j=1}^{m} < K_i y_{is}$ , for i ∈ 1, 2, 3… $n$
- Each plant i of capacity s is either open or closed.
  Constraint 3
  $Y_{is} \in \{1, 0\},\ for\ i = 1, 2, 3 \ldots n$
  $for\ s \in \{high,\ low\}$

- Delivery lead time should not exceed the maximum storage time of the product.
  Constraint 4
  $T_{ij} \leq S \ for\ i \in 1, 2, 3 ... n$
  $for\ j \epsilon 1, 2, 3 ... m$
  $S \epsilon\ storage\ time$

### 3.5 Objective Function

$$\sum_{i=1}^{n} f_{is} y_{is} + \sum_{i=1}^{n} \sum_{j=1}^{m} C_{ij} X_{ij}$$

The goal function described above reduces the overall cost of setting up and running the network (production plus variable costs).

The resulting solution identifies the factories that must remain operational and the type of production capacity that it will have.

## 4.Methodology

### 4.1 Linear Programming - Simplex method

The linear programming problem can be solved using the simplex method by identifying a workable solution,and creating solutions that lead to an ideal solution through an iterative process. The simplex technique tests whether the value of the objective function is optimal by starting at a feasible point.

If not, the process proceeds at a more advantageous location. It is claimed to be superior because, at a new point, the objective function's value is frequently very close to ideal. If this new point doesn't offer the best value, we go through the process again. If there is an ideal value, it will ultimately be produced through the simplex approach.

We made use of the Python PuLP package. The COIN-OR Foundation maintains PuLP, a Python modelling framework for linear (LP) and integer programming (IP) issues (Computational Infrastructure for Operations Research)..

### 4.2 Introduction to Greedy Algorithm

A greedy algorithm gradually develops a particular candidate solution. A greedy algorithm is so called because of how it selects from among the various ways to augment the present partial answer. The many options are often ranked according to some criterion, and the best option based on this criterion is selected. As a result, the algorithm constructs the solution by constantly choosing the action that at the time seems to hold the most promise. Even though there are many issues for which greedy tactics don't yield the best results, when they do, they frequently wind up being highly effective.

#### 4.2.1 Pseudo code used

1. The algorithm will first select plants so that their combined output capacity is either equal to or larger than their combined net demand.
2. Then it will proceed to a random city.
3. The least expensive criteria will next be used to determine the final market to city combination (greedy selection).
4. Once all market to city combinations have been discovered, the procedure will be repeated until all demand centres' demands have been met. At that point, it will move on to step 2 once more.
5. The total cost will be determined after all possible combinations have been found.

## 5.Numerical Analysis

Five plants and five demand centres made up the supply chain that was one of the model's inputs.

Below are listed the demands at each DC, monthly plant capacities, monthly fixed plant operating expenses, production and distribution costs per thousand units.

| Supply city | Demand city Variable cost (Rs/unit) | | | | | Monthly capacity of production (Ki) | | Monthnly fixed cost ($f_{is}$)* | |
|---|---|---|---|---|---|---|---|---|---|
| | D 1 | D 2 | D 3 | D 4 | D 5 | H | L | H | L |
| P1 | 12 | 12 | 12 | 12 | 12 | 1.5k | 500 | 9500 | 6500 |
| P2 | 13 | 13 | 13 | 13 | 13 | 1.5k | 500 | 7200 | 4800 |
| P3 | 10 | 10 | 10 | 10 | 10 | 1.5k | 500 | 9100 | 6230 |
| P4 | 8 | 8 | 8 | 8 | 8 | 1.5k | 500 | 4730 | 3230 |
| P5 | 5 | 5 | 5 | 5 | 5 | 1.5k | 500 | 6160 | 2110 |

Table 3

Regarding the output capacity of the factories, H stands for high capacity and L for low capacity.

The delivery lead time in days from factory i to market j is displayed in the table below.

| Supply city | Demand City | | | | |
|---|---|---|---|---|---|
| | D1 | D2 | D3 | D4 | D5 |
| P1 | 0 | 10 | 8 | 2 | 32 |
| P2 | 10 | 0 | 28 | 8 | 12 |
| P3 | 8 | 28 | 0 | 10 | 14 |
| P4 | 2 | 8 | 10 | 0 | 18 |
| P5 | 32 | 12 | 14 | 18 | 0 |

Table 4

Assuming a 30-day maximum transportation duration.
The company's manager must choose the locations of each facility and the kind of production capacity that each will have. We used Python to perform both strategies in search of the best outcome. We also took time constraints into consideration while implementing, initially without them and then with them.
Different demand scenarios for various countries are shown in table no. 4. These details served as inputs for the provided numerical example.
The quantity demanded is specified in terms of units.

| **Scenarios** | **S1** | **S2** | **S3** | **S4** |
|---|---|---|---|---|
| P1 | 160K | 200K | 1900K | 239K |
| P2 | 145K | 200K | 75K | 0 |
| P3 | 1700K | 800K | 2000K | 2800K |
| P4 | 90K | 190K | 180K | 550K |
| P5 | 2800K | 0 | 550K | 0 |

**Table 4**

**6.Result and Discussions**

The total cost using LP, the total cost using LP with added time constraint, the total cost using the greedy algorithm, and the total cost using the greedy algorithm with added time constraint are all compared in Table No. 5 for all the cases.
Tables 6 and 7 illustrate the best positions determined using the LP and greedy algorithms, respectively.
The findings from the above table clearly demonstrate how, in contrast to solutions derived via linear programming, the optimal solution determined using the greedy approach always provides us a near optimal solution.
Additionally, both algorithms display an indication of overall cost when time limits are included.

| **Scenariois** | **Total cost using LP*** | **Total cost using LP with time constraints*** | **Total cost using greedy algorithm*** | **Total cost using greedy with time constraint*** |
|---|---|---|---|---|
| **S1** | 92981K | 99406K | 103237K | 107371K |
| **S2** | 30786K | 30916K | 35643K | 35136K |
| **S3** | 70954K | 74617K | 80310K | 79872K |
| **S4** | 70320K | 78793 | 75575K | 85896K |

**Table 5**

***all the values are calculated in rupees**

| **Supply city** | **With time constraint** | | **Without time constraint** | |
|---|---|---|---|---|
| | **L** | **H** | **L** | **H** |
| **P1** | 1 | **1** | **0** | **1** |
| **P2** | **0** | **0** | **0** | **0** |
| **P3** | **0** | **1** | **0** | **1** |
| **P4** | **0** | **1** | **1** | **0** |
| **P5** | **1** | **0** | **0** | **1** |
| **Total cost** | **99406k*** | | **92981k*** | |

**Table 6**

***calculated in rupees**

| **Supply city** | **With time constraint** | **Without time constraint** |
|---|---|---|

| | L | H | L | H |
|---|---|---|---|---|
| **P1** | **0** | **1** | **0** | **1** |
| **P2** | **0** | **0** | **0** | **0** |
| **P3** | **0** | **1** | **0** | **1** |
| **P4** | **1** | **1** | **1** | **0** |
| **P5** | **1** | **0** | **1** | **1** |
| **Total cost** | **107371k*** | | **103237k*** | |

**Table 7**
***calculated in rupees**

## 7.Conclusions

In the suggested article, we took into account a supply chain network made up of five cities and five demand centres. Finding the best places to open a plant so as to reduce total cost was the aim of the proposed task. Additionally, it provided data on whether the facility was running at high or low capacity.

The data was initially optimised without a time limit, and then a time limit was introduced to improve the model.

Additionally, both models provide us the freedom to simulate various scenarios that affect operational and financial aspects.

After analysing all the data, it was determined that the greedy method was never as effective as the linear programming algorithm.

Additionally, both algorithms demonstrated an increase in cost when time constraints were added.

## 8. References

**[1]**
**https://www.samirsaci.com/supply-chain-optimization-with-python/**
**[2]**
**https://www.academia.edu/5714800/Supply_Chain_Network_Optimization_Using_New_Hybrid_PSO_Optimizer**