Distribution Network Optimization for Marico LTD

Aditya Satpute
Ian Heggen
Bingning (Icey) Xue
Shriya Yegalapati

05/2024

Table of Contents

Introduction

1. Background on Marico Ltd.

Marico Ltd. is a prominent fast-moving consumer goods (FMCG) conglomerate operating in India and 25 other countries across Asia and Africa. The company is well-known for its diverse range of products, including value-added hair oils, coconut natural oil, and edible oil, which collectively contribute to over 90% of its annual revenue of approximately ₹70 billion. Key product lines such as Parachute, Coco Soul, and Saffola have established Marico as a market leader in these segments.

2. Overview of the current distribution network

Despite its robust market presence, Marico has been grappling with significant challenges in its distribution network. The company's logistics and planning division has faced issues of high inventory levels coupled with low service levels across its 25 warehouses spread throughout India. These inefficiencies have been further exacerbated by the supply chain disruptions caused by the COVID-19 pandemic, necessitating a thorough review and optimization of the distribution network.

In response to these challenges, Marico's logistics and planning co-heads, Arun Parekh and Naren Shah, have initiated a comprehensive evaluation of the company's current distribution design. They aim to identify the root causes of the inefficiencies and propose a strategic plan to enhance service levels while reducing overall distribution costs. This task has been entrusted to Alia Gill, a talented intern with expertise in supply chain optimization.

3. Objectives

The primary objective of this case study is to analyze Marico's distribution network and recommend an optimized solution that addresses the high inventory and low service levels. By leveraging advanced optimization techniques such as Mixed Integer Linear Programming (MILP) and Network Flow Models, this study seeks to develop a more efficient and cost-effective distribution strategy for Marico Ltd. The findings and recommendations from this study will be crucial for the company to maintain its competitive edge in the FMCG sector while ensuring optimal service delivery to its customers.

Problem Statement

Marico faces significant challenges within its distribution network. The primary issues are high inventory levels and low service levels across its 25 warehouses located throughout India. Despite maintaining revenue levels and even seeing improvements in certain territories during the COVID-19 pandemic, the company struggles with inefficiencies in its logistics and planning division. These inefficiencies are further complicated by the need to balance cost reduction with maintaining or improving service levels to distributors and end customers.

The specific problems that Marico faces include:

1. **High Inventory Levels:** Warehouses are holding excessive stock, leading to increased holding costs and potential obsolescence.
2. **Low Service Levels:** Despite high inventory, service levels are not meeting expectations, indicating issues in inventory management and distribution practices.
3. **Bullwhip Effect:** Variability in customer demand is causing significant fluctuations in inventory levels upstream in the supply chain.
4. **Inaccurate Demand Forecasting:** The forecasting models are not accurately predicting demand, resulting in overstocking or stockouts.
5. **Inefficient Warehouse Management:** The current warehouse locations and management practices are not optimized for cost-effective and timely distribution.
6. **High Freight Costs:** Transportation costs are high, partly due to suboptimal routing and scheduling.

Objective Function & Constraints

The goal is to minimize the total cost from factories to warehouses and then from warehouses to distributors, which includes primary freight costs (PFC), secondary freight costs (SFC), and fixed costs (FC). The objective function can be expressed as:

Objective Function:

$$Minimize\ Z = \sum_{i\in I}\sum_{j\in J}\sum_{p\in P} ((\text{Primary Freight Intercept})_i + \text{Primary Freight Coefficient}_i \times d_{ij}) \times x^p_{ij}$$

$$+ \sum_{j\in J}\sum_{k\in K}\sum_{p\in P} ((\text{Secondary Freight Intercept})_j + \text{Secondary Freight Coefficient}_j \times d_{jk}) \times y^p_{jk}$$

$s.t.$

$x^p_{ij} \leq MZ_j \quad \forall\, i\in I, j\in J, p\in P$ (Capacity Constraints)

$y^p_{jk} \leq MZ_j \quad \forall\, j\in J, k\in K, p\in P$

$\sum_{j\in J} y^p_{jk} = Demand_{kp} \quad \forall\, k\in K, p\in P$ (Demand @ Distributor)

$\frac{\sum j\in J \sum k\in K \sum p\in P\ x\ d_{jk}\ x\ y_{jkp}}{Total\ Demand} \leq 500(km)$ (Distance to ensure 2-day delivery)

$\sum_i x^p_{ij} = \sum_l y^p_{jk}$ (Flow Conservation)

$\sum_j Z_j = [22, 24]$ (# of Operable Warehouses)

$x^p_{ij}, y^p_{jk} \geq 0$ (Non-negative Constraint)

Where:

- $d_{ij:}$ distance from factory i to warehouse j
- d_{jk}: distance from warehouse j to distributor k
- demand$_{kp}$: demand of product p at distributor k
- $W_i =$ Binary variable indicating whether a warehouse i is open

- Z_j = binary variable where $Z_j = 1$ if warehouse j is open; otherwise 0
- M: a large constant used for modeling logical constraints

Decisions Variables:

- x^p_{ij}: continuous variable representing the amount of product p shipped from factory i to warehouse j
- y^p_{jk}: continuous variable representing the amount of product p shipped from warehouse j to distributor k
- Z_j: binary variable where $Z_j = 1$ if warehouse j is open; otherwise 0

Breakdown of the Functions

Objective Function

1. Primary Cost:

$$\sum_{i\in I}\sum_{j\in J}\sum_{p\in P} ((\text{Primary Freight Intercept})_i + \text{Primary Freight Coefficient}_i \times d_{ij}) \times x^{p}_{ij}$$

The primary freight cost is a critical component in Marico's overall distribution expenses. This cost can be modeled using a linear equation $y = mx + c$ where y denotes the total freight cost in rupees per kilogram, m is the cost per kilogram per kilometer, x is the distance in kilometers, and c is the intercept representing the base freight cost per kilogram.

Each plant location has specific intercept and slope values, which reflect the base cost and the incremental cost per kilometer, respectively. For instance, the Sanand plant has an intercept of ₹2.85 and a slope of 0.0050 ₹/km/kg, indicating that for every kilometer, the freight cost per kilogram increases by ₹0.0050. Other plants have their unique intercepts and slopes, allowing for precise cost modeling.

By integrating these linear cost equations into our analysis, we can accurately estimate primary freight expenses per unit over various distances. This calculation is crucial for evaluating and optimizing Marico's distribution network, influencing decisions on warehouse placements, routing efficiencies, and overall logistics strategies.

2. Secondary Cost:

$$\sum_{j \in J} \sum_{k \in K} \sum_{p \in P} ((\text{Secondary Freight Intercept})_j + \text{Secondary Freight Coefficient}_j \times d_{jk}) \times y^p_{jk}$$

The cost of secondary freight for Marico's distribution network is determined by the distance traveled and varies across different regions. The relationship between the distance and the cost of secondary freight can be represented by linear equations in the form of $y = mx$, where y represents the total cost in rupees per kilogram, m represents the cost per kilogram per kilometer, and x represents the distance in kilometers.

Each region's specific linear equation captures the cost dynamics between distance and secondary freight. The coefficient m signifies how much the freight cost per kilogram increases with each kilometer.

For example, in the North East region, the equation $y = 0.0145x + c$ implies that for every kilometer traveled, the secondary freight cost per kilogram rises by 0.0145 rupees. Different regions have their unique coefficients and intercepts, which tailor the cost calculation to their specific conditions.

Incorporating these linear equations into our analysis allows us to estimate secondary freight costs over various distances and calculate the total transportation expenses for Marico's distribution network. This evaluation helps us understand how distance affects overall costs, guiding decisions on routing strategies and logistics optimization for each region.

Constraints

1. Capacity Constraints: x^p_{ij} , y^p_{jk} $\leq MZ_j$

Capacity constraints ensure that the amount of goods transported does not exceed the storage capacity of the warehouses. This constraint originates from the necessity to avoid overloading warehouses, which would lead to inefficiencies and higher costs. In Marico's case, it was essential to define these limits to maintain optimal warehouse operations and manage inventory effectively.

2. Demand @ Distributor: $\sum_{j \in J} y^{p}_{jk} = Demand_{kp}$

The demand constraint ensures that the distribution network meets the demand at each distributor location. This originates from the need to fulfill customer orders accurately and timely. Marico's objective was to match the supply from warehouses to the demand at each distributor, ensuring customer satisfaction and maintaining service levels.

3. Distance to ensure 2-day delivery: $\frac{\sum j \in J \sum k \in K \sum p \in P \; x \; d_{jk} \; x \; y_{jkp}}{Total\ Demand} \leq 500(km)$

The constraint on distance ensures that goods can be delivered within two days, typically modeled as a maximum distance of 500 kilometers. This originates from Marico's service level promise to deliver products within a specified timeframe. Ensuring warehouses are within a 500 km radius of their served distributors helps maintain this delivery commitment.

4. Flow Conservation: $\sum_{i} x^{p}_{ij} = \sum_{l} y^{p}_{jk}$

Flow conservation constraints ensure that the total inflow of goods to a warehouse equals the total outflow. This originates from the need to balance supply and demand within the network, ensuring that no warehouse is a bottleneck. For Marico, this was essential to maintain an efficient distribution system where goods are continuously moving through the network without accumulating excessively at any point.

5. # of Operable Warehouses: $\sum_{j} Z_{j} = [22, 24]$

The constraint on the number of operable warehouses restricts the total number of warehouses that can be open at any time. In this scenario, we choose between 22 or 24 warehouses.

6. Non-negative Constraint: $x^{p}_{ij}, y^{p}_{jk} \geq 0$

The non-negative constraint ensures that all decision variables, such as the amount of goods transported, are non-negative. This is a standard constraint in optimization models to reflect the reality that negative quantities of goods or costs are not possible.

Optimization Approaches

1. Data Preparation

Except for the data directly included in the material and Prof's announcement, additionally, we calculated zone centroids using the mean of the coordinates of all the warehouses in each zone:

Distributor	Zone	Latitude	Longitude
D1	Central	22.84090	80.12867
D2	East	23.64368	86.66461
D3	North	32.38277	76.16827
D4	North Central	26.94736	78.30360
D5	North East	26.36320	93.34935
D6	North West	22.85783	72.76833
D7	South East	13.86047	80.79221
D8	South West	14.05934	76.75621
D9	West	19.95136	76.78152

2. Choice of Model

To address the challenges in Marico Ltd.'s distribution network, we employed a combination of optimization techniques to achieve the lowest possible cost given all the constraints, including Mixed Integer Linear Programming and Network Flow Models.

1. Mixed Integer Linear Programming (MILP)

Mixed Integer Linear Programming extends Linear Programming by allowing some variables to be integers, which is particularly useful for modeling decision variables that are naturally discrete, such as the opening or closing of warehouses. MILP was instrumental in addressing Marico's issues with warehouse management. By incorporating binary variables to represent whether a warehouse is open or closed, MILP helped in determining the optimal number of warehouses that should remain operational to minimize costs while meeting service level requirements.

2. Network Flow Models

Network Flow Models are used to represent and optimize the flow of goods through a distribution network. These models are particularly effective in handling complex supply chain problems involving multiple sources, destinations, and intermediary nodes (warehouses). For Marico, network flow models were used to analyze and optimize the routes and schedules for transporting goods from factories to warehouses and from warehouses to distributors. This approach helped in reducing transportation costs and ensuring timely delivery.

Conclusion and Discussion

Conclusion

Our analysis concluded that maintaining 22 warehouses, strategically located to balance cost and service levels, was the optimal solution. Those opened warehouses are: W6, W8, W9, W10, W11, W12, W13, W14, W15, W16, W17, W18, W19, W20, W22, W25, W29, W41, W42, W46, W47, W48. The optimized network achieved a minimum freight cost of **₹5,702,969.10**, ensuring efficient and cost-effective distribution operations.

As the overall freight cost remains constant, this is still the better option as we are foregoing the additional costs that will be incurred opening additional warehouses, shipping inventory to the new warehouses, and obtaining talent.

Here are the distribution network we generated for Marico Ltd.:

- Plant to Warehouse Shipments:

	Plant	Warehouse	Product	Shipment Quantity	Distance
0	P1	W18	VAHO	18166	2157.17
1	P1	W19	VAHO	32294	1268.75
2	P1	W25	VAHO	20184	1176.09
3	P1	W29	VAHO	32294	1803.26
4	P1	W41	VAHO	28258	636.61
5	P1	W47	VAHO	6055	145.02
6	P2	W6	VAHO	2018	258.57
7	P3	W6	Edible Oil	2523	127.00
8	P3	W19	Edible Oil	40368	443.01
9	P4	W8	Edible Oil	50460	85.54

10	P4	W11	Edible Oil	27753	207.08
11	P4	W12	Edible Oil	22707	527.82
12	P4	W25	Edible Oil	25230	566.72
13	P4	W29	Edible Oil	40368	833.73
14	P4	W41	Edible Oil	35322	1136.94
15	P4	W47	Edible Oil	7569	1883.84
16	P5	W6	CNO	3869	2348.92
17	P5	W11	CNO	42555	1269.86
18	P5	W22	CNO	61898	1566.44
19	P6	W18	CNO	34817	168.03
20	P6	W42	CNO	77372	820.56
21	P7	W20	CNO	61898	65.12
22	P7	W41	CNO	54160	1419.92
23	P7	W46	CNO	38686	1048.76
24	P7	W47	CNO	11606	2137.39
25	P8	W8	VAHO	40368	481.07
26	P8	W16	VAHO	22202	128.29

- Warehouse to Distributor Shipments:

	Warehouse	Distributor	Product	Shipment Quantity	Distance
0	W6	D3	CNO	3869	53.35

1	W6	D3	Edible Oil	2523	53.35
2	W6	D3	VAHO	2018	53.35
3	W8	D9	Edible Oil	50460	90.37
4	W8	D9	VAHO	40368	90.37
5	W11	D6	CNO	42555	152.34
6	W11	D6	Edible Oil	27753	152.34
7	W12	D8	Edible Oil	22707	294.24
8	W16	D6	VAHO	22202	104.61
9	W18	D8	CNO	34817	160.44
10	W18	D8	VAHO	18166	160.44
11	W19	D4	Edible Oil	40368	91.39
12	W19	D4	VAHO	32294	91.39
13	W20	D7	CNO	61898	185.68
14	W22	D4	CNO	61898	247.59
15	W25	D1	Edible Oil	25230	61.34
16	W25	D1	VAHO	20184	61.34
17	W29	D7	Edible Oil	40368	153.26
18	W29	D7	VAHO	32294	153.26
19	W41	D2	CNO	54160	69.46
20	W41	D2	Edible Oil	35322	69.46

21	W41	D2	VAHO	28258	69.46
22	W42	D9	CNO	77372	150.39
23	W46	D1	CNO	38686	163.98
24	W47	D5	CNO	11606	35.46
25	W47	D5	Edible Oil	7569	35.46
26	W47	D5	VAHO	6055	35.46

	Demand <= 250km	
Region	**Current**	**Suggested**
Central	100%	NA
East	100%	63%
North	100%	73%
North Central	66.66667%	NA
North East	100%	NA
North West	100%	NA
South East	50%	NA
South West	66.66667%	NA
West	66.66667%	79%

	Demand <= 250km	
Region	**Current**	**Suggested**
Central	100	NA
East	100	63%
North	100	85%
North Central	50	NA
North East	100	NA
North West	100	NA
South East	50	NA
South West	75	NA
West	66.666667	93%

Discussion

While opening 24 warehouses might potentially improve service levels by reducing the average transportation distance and meeting some regional demands more effectively, the additional costs associated with opening and operating two more warehouses outweigh these benefits. The freight cost remains constant, and the marginal benefits do not justify the additional expenses.

It should be noted we provided a large value in the model (10,000,000,000,000) that essentially doesn't not impose the capacity constraint since we did not have any values. There is also no constraint on plant production capacity. Both of these should be considered as next steps for the company as they refine the model to enhance the optimization of freight cost reduction.

Appendix

1. Data

- Warehouses Location

Warehouse	Latitude	Longitude	City	State
W1	27.59137	76.43122	Roondh Rampur	Rajasthan
W2	26.85383	84.19225	Amwas Khas	Uttar Pradesh
W3	17.20569	81.53633	Rajahmundry	Andhra Pradesh
W4	28.56098	80.49032	Sumer Nagar	Uttar Pradesh
W5	29.25725	77.91551	Bhoomma	Uttar Pradesh
W6	31.91242	76.05651	Kasba	Himachal Pradesh
W7	13.02118	78.3928	Gennerahalli	Karnataka
W8	20.60115	76.2612	Taroda	Maharashtra
W9	23.03439	88.81676	Bangaon	West Bengal
W10	27.33021	95.48732	Lakhipathar	Assam
W11	21.86046	73.78392	Panchmuli	Gujarat
W12	16.27019	75.24948	Bagalkot	Karnataka
W13	17.82108	78.88491	Bhongir	Telangana
W14	26.55461	81.66674	Para	Uttar Pradesh
W15	22.37411	83.58641	Raigarh	Chhattisgarh
W16	23.52274	73.49236	Behdaj	Gujarat
W17	9.60788	77.85228	Moolippatti	Tamil Nadu
W18	12.62172	76.88276	K. Gowdagere	Karnataka
W19	27.47615	79.01114	Khiriya Nagar Shah	Uttar Pradesh
W20	12.52546	79.76196	Kancheepuram	Tamil Nadu
W21	20.41729	86.45314	Parakula	Odisha
W22	24.75631	78.74415	Gangasagar	Uttar Pradesh
W23	24.09555	87.79077	Karkaria	West Bengal
W24	25.83937	86.54676	Parari	Bihar

W25	23.28316	80.48695	Sakari	Madhya Pradesh
W26	23.19029	71.02871	Kutch	Gujarat
W27	24.82048	93.59155	Tamenglong	Manipur
W28	15.69152	78.36632	Nandyal	Andhra Pradesh
W29	14.54547	79.55849	Bandarupalli	Andhra Pradesh
W30	19.5598	73.6301	Hinglud	Maharashtra
W31	25.35598	76.39305	Tisaya	Rajasthan
W32	11.67809	92.71646	Port Blair	Andaman and Nicobar
W33	17.70758	82.50955	Anakapalle	Andhra Pradesh
W34	22.5318	77.09682	Dewas	Madhya Pradesh
W35	34.06834	74.79684	Batmaloo	Srinagar
W36	21.32824	82.51482	Dewgaon	Chhattisgarh
W37	23.26714	78.21346	Raisen	Madhya Pradesh
W38	24.04213	82.42423	Suhira	Madhya Pradesh
W39	27.6641	74.46837	Nagaur	Rajasthan
W40	25.39387	84.01259	Kasimpur	Bihar
W41	23.0816	86.36763	Rupapatia	West Bengal
W42	18.64162	77.13884	Wanjarwada	Maharashtra
W43	11.2273	76.97794	Coimbatore	Tamil Nadu
W44	18.83478	74.83712	Mandavgan	Maharashtra
W45	10.5946	79.75781	Venmanacheri	Tamil Nadu
W46	21.36683	80.17494	Purgaon	Maharashtra
W47	26.56603	93.07447	Deusur Sang	Assam
W48	14.32428	76.49978	Chitradurga	Karnataka
W49	26.73609	91.24404	Baksa	Assam
W50	20.70395	78.64692	Wardha	Maharashtra
W51	25.40305	73.72323	Kaklawas	Rajasthan
W52	31.16756	77.65145	Shimla	Himachal Pradesh
W53	23.0597	76.57799	Kalyanpura	Madhya Pradesh

- Plants Location

Plant	Latitude	Longitude	City	State
P1	26.1158	91.7086	Guwuhati	Assam
P2	30.3165	78.0322	Dehradun	Uttarakahand
P3	30.9578	76.7914	Baddi	Himachal Pradesh
P4	21.0077	75.5626	Jalgaon	Maharashtra
P5	10.7976	76.743	Kanjikode	Kerala
P6	11.2746	77.5827	Perundara	Tamil Nadu
P7	11.9416	79.8083	Pondicherry	Tamil Nadu
P8	22.992	72.3773	Sanand	Gujarat

- Distributors Location

Distributor	Zone	Latitude	Longitude	Net Demand
D1	Central	22.8409	80.12867	84100
D2	East	23.64368	86.66461	117740
D3	North	32.38277	76.16827	8410
D4	North Central	26.94736	78.3036	134560
D5	North East	26.3632	93.34935	25230

Code

Objective Setup

```python
## Model initialization

model = ConcreteModel()

# Sets
model.I = Set(initialize = [p for p in plant_df['Plant']])  # Plants
model.J = Set(initialize = [w for w in warehouse_df['Warehouse']])  # Warehouses
model.K = Set(initialize = [d for d in zone_df['Distributor']])  # Distributors
model.P = Set(initialize = ['CNO', 'Edible Oil', 'VAHO'])  # Products

# Parameters
model.d_ij = Param(model.I, model.J, initialize = {(i, j): P2W_df.loc[(P2W_df['Plant'] == i) & (P2W_df['Warehouse'] == j), 'Distance'].values[0]
                                                  for i in model.I for j in model.J})  # Distance from Plants to Warehouses
model.d_jk = Param(model.J, model.K, initialize = {(j, k): W2D_df.loc[(W2D_df['Warehouse'] == j) & (W2D_df['Distributor'] == k), 'Distance'].values[0]
                                                  for j in model.J for k in model.K})  # Distance from Warehouses to Distributors
#model.demand_kp = Param(model.K, model.P, initialize = {(k, p): demand[k][p] for k in model.K for p in model.P})  # Demand at Distributors
model.demand_kp = Param(model.K, model.P, initialize = {(k, p): zone_df.loc[zone_df['Distributor'] == k][p].values[0] for k in model.K for p in model.P}, mutable = True)

model.FactoryCanProduce = Param(model.I, model.P, initialize = {(i, p): plant_product[i][p] for i in model.I for p in model.P}, within = Binary)

# Primary freight costs (Plant to Warehouse)
model.primary_intercept = Param(model.I, model.J, initialize = {(i, j): P2W_costs.loc[(P2W_costs['Plant'] == i) & (P2W_costs['Warehouse'] == j), 'Intercept'].values[0]
                                                               for i in model.I for j in model.J}, mutable = True)
model.primary_coefficient = Param(model.I, model.J, initialize = {(i, j): P2W_costs.loc[(P2W_costs['Plant'] == i) & (P2W_costs['Warehouse'] == j), 'Coefficient'].values[0]
                                                                 for i in model.I for j in model.J}, mutable = True)

# Secondary freight costs (Warehouse to Distributor)
model.secondary_freight = Param(model.J, model.K, initialize = {(j, k): W2D_costs.loc[(W2D_costs['Warehouse'] == j) & (W2D_costs['Distributor'] == k), 'Secondary Freight'].values[0]
                                                               for j in model.J for k in model.K}, mutable = True)

model.numWarehouses = Param(initialize = 22)


## Total demand for each product
total_demand = sum(zone_df.loc[zone_df['Distributor'] == k][p].values[0] for k in model.K for p in model.P)

model.TotalDemand = Param(initialize = total_demand)


# Decision Variables
model.x = Var(model.I, model.J, model.P, within=NonNegativeReals)  # Quantity shipped from plant i to warehouse j of product p
model.y = Var(model.J, model.K, model.P, within=NonNegativeReals)  # Quantity shipped from warehouse j to distributor k of product p
model.z = Var(model.J, within = Binary)  # Binary variable for whether warehouse j is open
```

```python
## Objective
def objective_rule(model):
    # Calculate primary shipping costs from plants to warehouses using intercept and coefficient
    primary_costs = sum((model.primary_intercept[i, j] + model.primary_coefficient[i, j] * model.d_ij[i, j]) * model.x[i, j, p]
                        for i in model.I for j in model.J for p in model.P)

    # Calculate secondary shipping costs from warehouses to distributors using the secondary freight rate
    secondary_costs = sum(model.secondary_freight[j, k] * model.d_jk[j, k] * model.y[j, k, p]
                          for j in model.J for k in model.K for p in model.P)

    return primary_costs + secondary_costs

# Set the objective
model.cost = Objective(rule = objective_rule, sense = minimize)
```

Constraints

1. Capacity Constraints: x^p_{ij} , y^p_{jk} $\leq MZ_j$

```python
## Production constraints as certain plants produce certain products

def plant_product_capability(model, i, j, p):
    return model.x[i, j, p] <= 1000000000 * model.FactoryCanProduce[i, p]  # Large upper bound to essentially make it non-binding

model.factoryProductCapability = Constraint(model.I, model.J, model.P, rule = plant_product_capability)
```

2. Demand @ Distributor: $\sum_{j \in J} y^p_{jk} = Demand_{kp}$

```python
## Demand must be fulfiled for each zone

def demand_fulfillment(model, k, p):
    return sum(model.y[j, k, p] for j in model.J) == model.demand_kp[k, p]

model.demandFulfillment = Constraint(model.K, model.P, rule = demand_fulfillment)
```

3. Distance to ensure 2-day delivery: $\frac{\sum j \in J \sum k \in K \sum p \in P \ x \ d_{jk} \ x \ y_{jkp}}{Total\ Demand} \leq 500(km)$

```
## Average distance <= 500 km for 2 day service

def average_distance(model):
    return sum(model.d_jk[j, k] * model.y[j, k, p] for j in model.J for k in model.K for p in model.P) <= 500 * model.TotalDemand
```

4. Flow Conservation: $\sum_{i} x^{p}_{ij} = \sum_{l} y^{p}_{jk}$

```
## Product outflow from warehouse = product inflow at warehouse
def flow_conservation(model, j, p):
    return sum(model.y[j, k, p] for k in model.K) == sum(model.x[i, j, p] for i in model.I)

model.flowConservation = Constraint(model.J, model.P, rule = flow_conservation)
```

5. # of Operable Warehouses: $\sum_{j} Z_{j} = [22, 24]$

```
## Constrain the number of operable warehouses

def warehouse_operation(model):
    return sum(model.z[j] for j in model.J) == model.numWarehouses

model.warehouseOperation = Constraint(rule = warehouse_operation)
```

6. Non-negative Constraint: $x^{p}_{ij}, y^{p}_{jk} \geq 0$