Maximization of Profit
in Dairy Industry

Gaurav Jain, Vihaan Thora

Indian Institute of Technology, Indore
Optimization Algorithm & Techniques - CS 357
Dr. Kapil Ahuja

December 2 2022

Contents

1 Abstract

Milk industry is an important illustration of supply chain management problems. The milk industry is based on the single input multiple output paradigm. At the same time, owing to the large demand of milk, industries have to purchase milk from multiple sources, which are located at different locations. All this makes it very hard for them to compute the ideal breakdown of the parameters that affect their final profit. This project aims to introduce a model that simplifies this problem for these industries.

2 Rationale

Milk industries usually purchase milk from multiple suppliers in a district to meet their daily production needs. These suppliers have a daily milk production limit based on the livestock they have, and may also sell their milk at different rates.

The factories then combine all of this milk. Here, we are assuming that the quality of milk across all the suppliers is kept similar, and we are treating the raw milk (one with all the initial elements such as fat) as one single product. We also consider the cost of shipping this milk from the supplier to an industry based on the distance between them.

The raw milk is then used to manufacture different types of items. All the items have a daily production limit and a pre-calculated milk requirement and production cost per unit. All the different items are then sold by the industry at different prices. There are specified minimum quantities associated with each type of item, in order to meet the daily requirement of the consumers for that item in the market.

Our goal is to determine the maximum daily profit the factory can earn and find out the exact breakdown of the different quantities of milk and items involved at each step.

3 Problem Statement

Consider the positive quantities,

- $n :=$ no. of milk suppliers
- $m :=$ no. of factories located at different places
- $t :=$ types of milk items

For the below definitions, we consider $i \in \{1 \dots n\}$, $j \in \{1 \dots m\}$, $k \in \{1 \dots t\}$

- $L_i :=$ maximum amount of milk that supplier i can supply
- $R_i :=$ per litre price charged by supplier i
- $D_{n \times m}$ where $D_{ij} :=$ distance between supplier i and factory j
- $Q_{m \times t}$, where $Q_{jk} :=$ maximum quantity of item k that can be manufactured by a factory j per day
- $A_k :=$ amount of milk required to produce per unit of item k
- $P_k :=$ per unit production cost of item k
- $S_k :=$ per unit selling price of item k
- $M_k :=$ minimum quantity of an item k to be manufactured by all factories combined

To compute the cost of delivering milk from the supplier to the factory we consider the cost function as

$$C(d, l) = C_0 d \left\lceil \frac{l}{l_0} \right\rceil$$

where,
$l :=$ amount of milk being transported,
$d :=$ distance,
$l_0 :=$ capacity of each truck, and
$C_0 :=$ normalizing constant

In this project, we wish to maximize the net profit of the company across all its factories.

4 Objective Function

To solve this problem, we must define two sets of variables -

1. $X_{n\times m}$, where X_{ij} := Amount of milk purchased (in litres) from supplier i by factory j.
2. $Z_{n\times m}$, where Z_{ij} := Number of trucks required to transport milk from supplier i to factory j.
3. $Y_{m\times t}$, where Y_{jk} := Quantity of item k manufactured by factory j

By using these variables, we now calculate the net profit produced by all factories. Thus, we can define our objective function $f(X, Y)$ as

$$max\ f(X,Y,Z) = \sum_{j=1}^{m}\left(\sum_{k=1}^{t}(S_k - P_k)Y_{jk} - \sum_{i=1}^{n}(R_i X_{ij} + C_0 D_{ij} Z_{ij})\right)$$

subject to constraints

1. $X_{ij} > 0\ \forall i \in \{1 \ldots n\}, \forall j \in \{1 \ldots m\}$
2. $Y_{jk} > 0\ \forall j \in \{1 \ldots m\}, \forall k \in \{1 \ldots t\}$
3. $Z_{ij} = \left\lceil \frac{X_{ij}}{l_0} \right\rceil\ \forall i \in \{1 \ldots n\}, \forall j \in \{1 \ldots m\}$
4. $\sum_{j=1}^{m} X_{ij} \leq L_i\ \forall i \in \{1 \ldots n\}$
5. $Y_{jk} \leq Q_{jk}\ \forall j \in \{1 \ldots m\}, \forall k \in \{1 \ldots t\}$
6. $\sum_{j=1}^{m} Y_{jk} \geq M_k\ \forall k \in \{1 \ldots t\}$
7. $\sum_{k=1}^{t} A_k Y_{jk} \leq \sum_{i=1}^{n} X_{ij}\ \forall j \in \{1 \ldots m\}$

4.1 Transformation of equations into vector form

To make the representation of the equations simpler, we convert the variables and constants into vectors.

1. $X_{nm\times 1} = \begin{bmatrix} X_{11} & X_{12} & \dots & X_{1m} & \dots & X_{nm} \end{bmatrix}^T$
2. $Y_{mt\times 1} = \begin{bmatrix} Y_{11} & Y_{12} & \dots & Y_{1t} & \dots & Y_{mt} \end{bmatrix}^T$
3. $Z_{nm\times 1} = \begin{bmatrix} Z_{11} & Z_{12} & \dots & Z_{1m} & \dots & Z_{nm} \end{bmatrix}^T$
4. $R_{nm\times 1} = \begin{bmatrix} R_1 & \dots & R_1 & R_2 & \dots & R_2 & \dots & Rn & \dots & Rn \end{bmatrix}^T$
   where each R_i is repeated m times.
5. $S_{mt\times 1} = \begin{bmatrix} S_1 & \dots & S_t & S_1 & \dots & S_t & \dots & S_1 & \dots & S_t \end{bmatrix}^T$
   where each $S_1 \dots S_t$ is repeated m times.
6. $P_{mt\times 1} = \begin{bmatrix} P_1 & \dots & P_t & P_1 & \dots & P_t & \dots & P_1 & \dots & P_t \end{bmatrix}^T$
   where each $P_1 \dots P_t$ is repeated m times.
7. $D_{n\times m} = \begin{bmatrix} D_{ij} \end{bmatrix}^T$

The objective function becomes -

$$max\ f(X, Y, Z) = S^T Y - P^T Y - R^T X - C_0 D^T Z$$

subject to constraints

1. $X > 0$
2. $Y > 0$
3. $\frac{X}{l_0} \leq Z < \frac{X}{l_0} + 1,\ Z \in \mathbb{N}$
4. $\sum_{j=1}^{m} X_{ij} \leq L_i\ \forall i \in \{1 \dots n\}$
5. $Y \leq Q$
6. $\sum_{j=1}^{m} Y_{jk} \geq M_k\ \forall k \in \{1 \dots t\}$
7. $\sum_{k=1}^{t} A_k Y_{jk} \leq \sum_{i=1}^{n} X_{ij}\ \forall j \in \{1 \dots m\}$

4.2 Formulation into standard LP

We wish to transform the problem into the standard form of LP, which is

$$min\ c^T x$$

subject to constraints

$$ax \le b,\ x \ge 0$$

Hence, we need to combine the three variables into a single vector **x**. Let $\alpha = (2nm + n + m + t + mt)$ and $\beta = (2nm + mt)$.

$$x = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}_{\beta \times 1}$$

We also need to club the weights associated with the variables into a single vector **c** and negate it to transform maximization problem to minimization.

$$c = \begin{bmatrix} R & P - S & C_0 D \end{bmatrix}^T$$

The constraint matrix **a** becomes

$$a = \begin{bmatrix} -\frac{I_{mn\times mn}}{l_0} & \mathbf{0} & I_{mn\times mn} \\ \frac{I_{mn\times mn}}{l_0} & \mathbf{0} & -I_{mn\times mn} \\ M_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & I_{mn\times mn} & \mathbf{0} \\ \mathbf{0} & -M_2 & \mathbf{0} \\ -M_3 & I_{mn\times mn} & M_4 \end{bmatrix}_{\alpha\times\beta}$$

where

$$M_1 = \begin{bmatrix} \mathbf{e}^T_{m\times 1} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{e}^T_{m\times 1} & \dots & \mathbf{0} \\ \dots & \dots & \dots & \dots \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{e}^T_{m\times 1} \end{bmatrix}_{n\times nm}$$

$$M_2 = \begin{bmatrix} I_{t\times t} & I_{t\times t} & \dots (m\ times) \end{bmatrix}_{t\times mt}$$

$$M_3 = \begin{bmatrix} I_{m\times m} & I_{m\times m} & \dots (n\ times) \end{bmatrix}_{m\times nm}$$

$$M_4 = \begin{bmatrix} A & \mathbf{0} & \mathbf{0} & \dots \\ \mathbf{0} & A & \mathbf{0} & \dots \\ \dots & \dots & \dots & \dots \\ \mathbf{0} & \mathbf{0} & \dots & A \end{bmatrix}_{m\times mt}$$

and

$$b = \begin{bmatrix} \mathbf{1}_{nm\times 1} \\ \mathbf{0}_{nm\times 1} \\ L \\ Q \\ -M \\ \mathbf{0}_{m\times 1} \end{bmatrix}_{\alpha\times 1}$$

Note that here, we have removed strict inequalities to make the solution process simpler. In real world scenario, we can make the value of the variable close the equality condition, to obtain an optimal solution.

5 Simplex Method

We implement the revised simplex method and the tableau simplex method to calculate the relaxed version of the problem i.e. without any integrality constraints.
The revised simplex method as opposed to the standard simplex method, does not recalculate the inverse of the basis matrix (B) in every iteration, rather it derives it from the old basis matrix, hence saving lot of computational complexity.

It is noted that, revised simplex method allows for greater computational efficiency especially for sparse matrices by enabling sparse matrix operations, which is the case here.

6 Cutting Planes Algorithm

The cutting plane method is commonly used for solving ILP and MILP problems to find integer solutions, by solving the linear relaxation of the given integer programming model, which is a non-integer LP model. A cutting plane algorithm is generally used to search for valid inequalities that cut-off the non-integer solutions in two cases, when the set of constraints in our integer programming model is too large, and when the inequality constraints in the original integer programming model are not sufficient to yield an integer solution.

The basic idea of the cutting plane method is iteratively refining the search region by introducing linear inequalities, known as cuts, maintaining the original feasible region. If the mathematical model is linear, an extreme or corner point in the feasible region can be the optimal solution, which may or may not be integer solution. If it is not integer, a linear inequality, in other words a cut, can be found to cut away a part of the feasible region, so as to separate the optimum solution. Therefore, a new separation problem is introduced and a cut is added to the relaxed LP model, which makes the existing non-integer solution no longer in the feasible region. This cutting process is repeated until the optimal solution found is also an integer solution. The steps of the cutting plane algorithm can be explained as follows:

1. The relaxed integer programming problem (the problem with continuous variables instead of discrete/integer variables) is solved.

2. Stop, if all variables in resulting solution have integer values, which means that it is the optimum.

3. Otherwise, generate a cut, that is, a constraint in the form of a linear inequality, which is satisfied by all feasible integer solutions.

4. Add this new constraint to the model, resolve the problem, and go back to step 2.

7 Branch and Bound Algorithm

This is a popular algorithm used to solve MILP problems. For this, we first relax our integer constraint to take any real value within the bounds, and then split the obtained minimizer into two ranges, one from the lower bound to the floor of the obtained value, and other from the ceil of the obtained value to the upper bound. Below is a description of this algorithm.

To start, we solve the relaxed version of the LP using the regular simplex method to get an upper bound. The solution can be integer or non-integer. If we get an integral solution then we don't need to branch that node anymore. We match the solution with the global max and update global max if necessary. If we get fractions in our solution, then we need to branch in hopes of getting a better integral solution. We can prune the branching process of a node in the following three ways.

- **Integrality** - the node has integer solutions.
- **Value** - the node has a value which is lesser than a value obtained at some other node (which is an integer and candidate solution).
- **Infeasibility** - the obtained solution lies in an infeasible region.

Branching as described above will create two sub problems at every step. Since, at every branch, we form a node based on the maximum (or most optimal solution) in that path, all the nodes present below, whose feasible regions are subsets of the parent node's feasible region, will have a lesser value than the parent. Hence, we can visualize the branch and bound tree as a **max heap**.

8 A Faster Compromise-Based Method

8.1 Limitations of Cutting Planes and Branch and Bound

1. Every cut of cutting planes algorithm ensures that the integer solutions of the previous problem are preserved. Thus, the complexity of determining the next cut using a standard algorithm (such as Gomory cuts) increases as the square of the iteration number. Hence, it involves running the cutting plane determining algorithm $O(nm^2)$ times, which is the square of the dimension of Z. And for every sub problem, we have to run a parallel simplex to find its minimizer. Hence, cutting planes method becomes infeasible for larger problems.

2. As mentioned in the above section, branch and bound involves increasing the number of problems by splitting a node into two at every step. Hence, the worst-case scenario would create exponential number of nodes to be evaluated. This would require us to run simplex exponential times. Taking an average real case example, the value of $n \times m$ can exceed well beyond a thousand, which means there are more than thousand components in our

integer vector Z, which can ultimately create a branch and bound tree deeper than a thousand levels. Hence, we would require to run simplex at least 2^{1000} times! This is something which is clearly not feasible.

8.2 Our method

Thus, to bring in realistic run times of the algorithm, we have introduced a case specific solution approach, which is described below.

Assuming the region encompassing all the milk farms and factories to be well populated. To bring integrality in the variable Z, we apply the transformation

$$Z_{ij} = \begin{cases} \lfloor Z'_{ij} \rfloor & \text{if } D_{ij} > d_{\max} \\ \lceil Z'_{ij} \rceil & \text{otherwise} \end{cases}$$

where,

$$d_{\max} = \max_{i,j} D_{ij}/2$$

Z' is the solution obtained by relaxing integer constraint

This ensures the notion that the factory buys more milk from the nearer supplier and less milk from the far away ones. Therefore the penalty incurred by empty trucks comes down and cost is minimized.

9 Generating Data

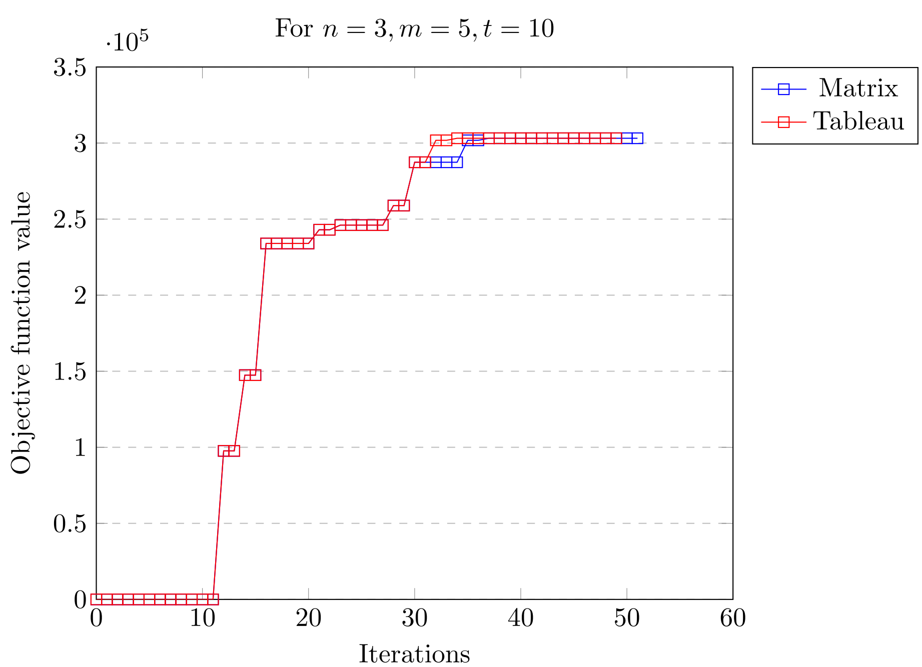
9.1 Input

All constant coefficient vectors are randomly generated numbers in a given range. The provided are ranges are based on realistic data to obtain relevant results. The distance matrix D is calculated by generating random points on a plane and then finding the distance between them using distance formula. All these values are then collectively dumped inside a **json** file, which is then read by our model as the input data. Here is an example of the input generated when we take n = x, m = y, t = z.

9.2 Output

Here is an example of the obtained output after solving the optimization problem.

10 Graph



11 Inferences

For constant value $n = 3, m = 5, t = 10$
Number of Iterations of Matrix method : 51
Run time for Matrix method : 0.017329 seconds
Number of Iterations of Tableau method : 49
Run time for Tableau method : 0.03851 seconds

Therefore, even though Tableau method converges in less iterations, the simplex method is twice as fast.

Due to randomly generated data sets, we infer that LP problem formed does not always gives a feasible region. In this case when we run our simplex, the non-negativity condition might get voided. There have also been seldom occurrences of unbounded simplex which do not converge, and stop after exhausting maximum number of iterations, leading to divergent solutions. Therefore, one might have to re-generate input data in those cases.

To remedy this, one might generate more practical random values. Some effort has already been applied, but there is always a scope for improvement.

References

[1] https://www.cs.upc.edu/~erodri/webpage/cps/theory/lp/milp/slides.pdf,
https://www.gurobi.com/resource/mip-basics/

[2] https://www.lindo.com/index.php/products/lingo-and-optimization-modeling

[3] https://in.mathworks.com/help/optim/ug/mixed-integer-linear-programming-algorithms.html

[4] https://github.com/mmolteratx/Simplex

[5] https://github.com/carlo98/mission-planning-optimization

[6] https://nptel.ac.in/courses/112106131