Network Flow Optimization

Aryan Pandhi
Cornell University

February 2020

1 Introduction

The problem consists of a jellybean manufacturing firm that has to produce a certain number of jellybeans based on the order it receives from its customers. The manufacturing starts on April 1^{st} (8 a.m.) and the firm wants its order to be completed by September 30^{th} (5 p.m.) to fulfil seasonal demand for Halloween. The jellybean manufacturing firm has 5 facilities with different manufacturing capabilities and storage constraints. The aim of this project is to design a distribution network that allows the optimal product flow through the facilities i.e. to divide the entire order received from the customers into internal work orders for each manufacturing facility such that the production is completed in the minimum amount of time.

2 The Problem Statement

The jellybean manufacturing firm is headquartered in Chicago, IL with manufacturing facilities in five locations spread throughout the mid-western region of the United States as shown in Figure 1.

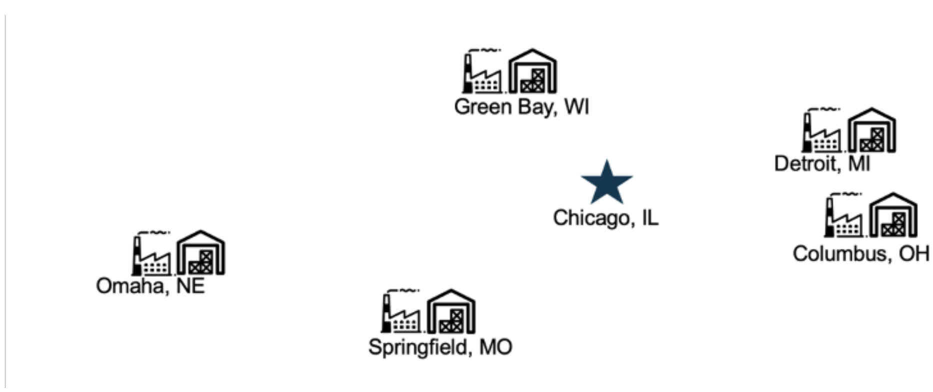

Figure 1: Locations of JB Manufacturing Headquarters and Manufacturing Facilities[1]

The jellybeans are produced in different colors, sizes, flavors and packaging methods. There are 40 different colors, 5 sizes, 12 flavors, and 2 packaging types (bag or box). The company receives an order bank consisting of 10,000 orders (order_bank.csv is the file containing the orders). Each order specifies a color, size, flavor, packaging type and the quantity of bags or boxes. A bag holds 0.25 pounds and a box holds 2.5 pounds. The goal is to fulfill the entire order bank received from the customers in minimum amount of time by splitting it into 5 internal work orders (one for each facility).

2.1 Three Stages of Manufacturing

There are three processes that the jellybeans go through in each facility. Initially, each manufacturing facility has a raw material inventory (RMI) store that holds jellybeans of various colors in containers. Each container contains jellybeans of the same color. The number of containers, the capacity of the containers, and the quantity and color of jellybeans in each container varies with the facility (rmi_inventory_level.csv contains the details for each facility).

2.1.1 Classifier

Each manufacturing facility has one classifier machine that splits the jellybeans from the containers in the RMI according to their sizes[2] (The jellybeans are split into different sizes simultaneously). The classifier separates all the jellybeans it gets as input simultaneously into different sizes. The classifier in each manufacturing utility operates at a different rate (classifier.csv contains the processing rates of the classifiers at different locations). Each RMI container of a particular color has a fixed percentage of sizes (classifier_split.csv contains the ratio of sizes for each color). The order in which the jellybeans are emptied from the RMI into the classifier is based on the internal work order for the particular facility.

The jellybeans that have been split up are stored in the pre-finish inventory (PFI) store. The PFI store stores jellybeans of the same color and size in a container. For simplicity of the model, the PFI store is assumed to have infinite storage.

2.1.2 Pre-finish Operation

After the jellybeans are split into the 5 sizes, the jellybeans of the desired size (according to the internal work order) go through the pre-finish operation. In the pre-finish operation, the desired flavor is added to the jellybeans (prefinish_data.csv contains large amounts of past data on the processing rates for different combinations of color and size).

[1] https://cdn.filestackcontent.com/RJTYRZznTm1x6c7Sq6OX?PobStatement_10-31-19_%20FINAL_INFORMS%202020%20StudentCompetition_ComprehensiveV3%20(1).docx

[2] Note: The classifier passes a quantity greater than the required amount for an order because a particular order in the internal work order is specific to a certain size

2.1.3 Packaging

The jellybeans are directly sent from the pre-finish operation to the packaging process. This is where the jellybeans are boxed or bagged based on the internal work order (packaging_data.csv contains large amounts of past data on the processing rates for different combinations of size and packaging type).

Figure 2: Process Flow[3]

Manufacturing Facility	Number of RMI Drums	Number of Pre-finish Operation Equipment	Number of Packaging Machines
Detroit, MI	40	2	1 bag, 1 box
Columbus, OH	30	3	2 bag, 1 box
Springfield, MO	50	1	1 bag, 1 box
Greenbay, WI	20	2	1 bag, 1 box
Omaha, NE	30	3	1 bag, 1 box

Table 1: Amount of Equipment and Drums at Each Operation and Inventory Store[4]

3 Methodology

3.1 Data Analysis

The distributions of the pre-finish operation and packaging processing rates had to be determined in order to create an efficient product flow.

prefinish_data.csv contains a thousand entries of processing rates for each location, size, and flavor. The density plots for a few combinations of these features were plotted on MATLAB's distribution fitter. The probability density function had a bell curve and most strongly resembled the gaussian distribution. Then the data for each combination of location, size,

[3]https://cdn.filestackcontent.com/RJTYRZznTm1x6c7Sq6OX?PobStatement_10-31-19_%20FINAL_INFORMS%202020%20StudentCompetition_ComprehensiveV3%20(1).docx

[4]https://cdn.filestackcontent.com/RJTYRZznTm1x6c7Sq6OX?PobStatement_10-31-19_%20FINAL_INFORMS%202020%20StudentCompetition_ComprehensiveV3%20(1).docx

and flavor was fit into a gaussian distribution using a python script that imported the SciPy package[5]. The mean and standard deviation for each combination was determined. The Kolmogorov Smirnov test was used to make sure that gaussian distribution was the right choice for each combination as it is not practical to view each distribution curve. The Kolmogorov Smirnov test determines how close two distributions are by calculating a p-value. The smallest p-value obtained for a combination was 0.12. This was still greater than the chosen significance value of 0.05.

The same process was repeated to determine the distribution of the packaging rates for each location, size and packaging type. The packaging data also followed a gaussian distribution and the Kolmogorov Smirnov test showed similar results.

prefinish_statistics.csv and packaging_statistics.csv contain the mean processing rate values, their standard deviations and their p-value statistics. It was observed that even though the mean values of the processing rates differed with different combinations (size and flavor for pre-finish and size and for packaging), the difference was very small per facility. In other words, the processing rate mainly depends only on the facility and the packaging rate on the facility and the packaging type. However, for the simulation, the values determined for each combination were used instead of the mean value for the facility.

3.2 Simulation Model

The simulation model for the entire production process was created using a Python script that imported a simulation modelling package called SimPy (simulation.py contains the code for the simulation model). The processing rates for the pre-finish and packaging operations were generated using a random number generator that used a gaussian distribution with the mean and standard deviation values found in the data analysis stage. The simulation model inputs the internal work orders for each facility and outputs the time each facility takes to process it.

The assumptions made for the model:

1. The PFI stores have infinite storage capabilities
2. Jellybeans cannot be transferred between the RMI of the facilities

3.3 Algorithm Design

3.3.1 Design Objectives

The objective of the algorithm is to process the entire order bank in as short a time span as possible. To minimize the time taken for the processing of the entire order it is important to know the variables that affect the total time taken when designing the product flow. Then smaller objectives need to be created that lead to the minimization of the entire process.

[5] prefinish_distribution_fit.py contains the script

To split the entire order bank, the total amount of demand for each type of color and size of jellybeans was found by adding up individual orders placed for that type. The flavor and packaging type were not considered at this stage because the processing rate of each of the machines in the facilities does not depend on the flavor and the packaging type was left out for simplicity of the procedure. Once the total demand for each type (color and size) of jellybeans was calculated, the process of creating internal work orders was split into two main processes:

1. Splitting up the total amount of each type of jellybeans into separate facilities.

2. Forming a sequence of orders (an internal work order) for each facility.

The main objective of forming the sequence of orders at each facility was to make sure that the jellybeans flow through them as fast as possible. However, the main objective of splitting the total amount was to make sure that each facility takes almost the same amount of time. This is because the total time taken is determined by the facility that takes the maximum amount of time.

The only resource that is considered in this problem is time. To minimize the total time taken, the following objectives need to be considered:

1. Processing each type of jellybeans at the facility that can do it in the shortest time

2. Minimizing the amount of time spent vacant by any machine at any facility

3. Minimizing the time taken by the facility that finishes last

The algorithms to split the order and creating internal work orders are designed in consideration with these objectives.

3.3.2 Time Approximations

To satisfy objective 1 and split the jellybeans according to how long they will take to be processed at the facilities a heuristic technique was used because it is not possible to predict the exact time that will be taken as it is a continuous flow and is dependent on their position in the sequence of orders. The amount of time the jellybeans would take at each facility was predicted using the following method of calculation:

Let $Rate(i, j)$ be the processing rate (in pounds per hour) of process i at facility j
Let $Amount(q)$ be the amount (in pounds) of jellybeans to be passed through the classifier for an order of quantity q
Let $N(i, j)$ be the number of machines for process i at facility j
Let C, Pr, Pa represent classifier, pre-finish and packaging respectively

For an order of q pounds of jellybeans at facility j:

$$Time(q, j) = \frac{Amount(q)}{Rate(C, j) \cdot N(C, j)} + \frac{q}{Rate(Pr, j) \cdot N(C, j)} + \frac{q}{Rate(Pa, j) \cdot N(C, j)}$$

The time was predicted by summing the time jellybeans would spend at each process in the facility.

This method was used to approximate the time that would be taken for the amount demanded of each combination of jellybeans (color and size) at each facility.[6]

3.3.3 Distributing the Order Bank

The classifier takes the required quantity of jellybeans of a particular color and splits them into their different sizes and stores them in the PFI store. The total order was iterated color wise and each color was distributed to facilities in the goal of sending as many jellybeans of the same color to a particular facility so that the excess jellybeans (of other sizes) in the PFI can be used for an order later in the sequence. This reduces the classifier time for the order later in the sequence as the classifier will have to split a smaller quantity of jellybeans. This helps with objective 2 as it reduces the amount of vacancy time of the pre-finish operation for the instances when it is done processing and is waiting for the output of the classifier.

The goals for distributing each color were to give a facility as much of a color and in different sizes such that proportion of the sizes is as close as possible to the proportion of the sizes in the RMI drums (classifier_split.csv contains the percentages). This would help use more of the already classified jellybeans in the PFI store and reduce the classifier time for the orders that use these jellybeans.

Initially, for a particular color, the total order has the amounts of jellybeans of each size that need to be produced. From these amounts of each size, certain quantities of each size are selected in such a way that the proportion of these quantities is equal to the proportion of sizes in the RMI drums (classifie_split.csv contains the percentages). These selected quantities are distributed to facilities. However, for each size, the quantity selected will be less than or equal to its total amount and all the color will not be completely distributed. When maximizing the quantities that can be selected of each size while maintaining the proportion, the total amount of one of the sizes will be completely used up. However, each of the other sizes may have some amount of jellybeans still to be distributed. So, after the first distribution, the percentage value of the size that was used up is divided among the other sizes that are left based on their percentage values. For e.g. if the size A that was used up had a percentage value of 23% and of the sizes B that is not used up had a percentage value of 18%, then the new percentage value of B will be $18 + \frac{18}{100-23} \cdot 23$. This makes sure that the percentage values of the remaining 4 sizes add up to 100%. Then the remaining amounts

[6] function approx_time() in total_order.py

are distributed to facilities according to these new percentages. This process is repeated 5 times to make sure all of the sizes are completely distributed.

The entire process of distributing the quantities of color and size to facilities takes place in the following way[7]:

```
For each color:
    For i from 1 to 5:
        determine the distribution amount of the sizes
        find the facilities with enough RMI for the entire amount
        If there exist facilities with enough RMI:
            assign the quantities to the facility that will cause the total
            time for the entire work order to be divided most equally
        Else:
            split it between the facilities in a way that the total time
            for the entire work order is divided most equally
        Endif
    EndFor
EndFor
```

The time taken to process the entire order was distributed as equally among the facilities as possible to satisfy objective 3. To determine the facility that will result in most equal time distribution when the quantities are assigned to it, an array of length 5 was kept and updated at each assignment. At each assignment, max(array) was determined for when the quantity would be assigned to each facility. The facility for which max(array) after assignment would be minimum is chosen.

The flavors and packaging type were assigned to the order of jellybeans that was already divided in the following way[8]:

```
For each combination of color, size, flavor, packaging type:
    determine total quantity demanded for the combination
    For each facility:
        If the facility has any amount of its order with the same color and size
        that has not been assigned flavor and packaging:
            assign flavor and packaging to as much amount as possible
            reduce the quantity left to be assigned by this amount
        Endif
    EndFor
EndFor
```

At the end of the assignment of flavor and packaging type, each facility had a quantity for each combination of jellybeans it had to produce.

[7]function distribute_total_order() in total_order.py

[8]function assign_flavor_package() in total_order.py

3.3.4 Creating the Internal Work Orders

For each facility, the quantities they had to produce had to be sequenced into a list of orders (also known as the Internal Work Order). To produce the internal work orders, the quantity of each combination (color, size, flavor, packaging) of jellybeans that each facility had to produce was set to be an order in its internal work order. These orders were divided into box packaging and bag packaging. The orders with bag packaging were sorted in descending order based on the time each combination would take. The combinations with box packaging were sorted in the same way. To form the final internal work order for each facility, the sequence of sorted bag orders were followed by the sequence of sorted box orders. csv files were created for each facility containing its entire internal work order[9].

The only way to minimize time while creating internal work orders was to satisfy objective 2 as much as possible. The internal work orders were created in this way because the bag packaging rates were less than the box packaging rates. The aim is to make sure each equipment at each facility is vacant for the least amount of time. If an order in the sequence takes more time than its predecessor then the time gap between two processes when the latter process takes less time is reduced.

4 Analysis of Results

The internal work orders produced were simulated 20 times and the mean time to process the entire work order was recorded. It was also observed that the simulation times do not vary much with more iterations i.e. they have a low standard deviation.

It takes an average of 3840.0 hours (160 days) to process the entire work order. The production ends on 7^{th} September at 8 a.m.

The algorithm was effective in completing the production of the entire work order within the given time frame. It finished the production 23 days in advance.

The problem statement was not the same as the original because certain assumptions were made due to the shortage of time and team members. It was assumed that jellybeans cannot be transferred between facilities prior to the production. Therefore, the model can be improved by allowing the flow of jellybeans and solving a constrained optimization problem. This is because there are shipping constraints such as the weight of shipments and there is an objective of minimizing total time to produce the jellybeans. Simpler alternative objectives could be created that cause the flow of jellybeans to facilities that are best at producing them, and transfer jellybeans such as to distribute the time it would take for the entire order as equally between facilities as possible.

It was also assumed that the PFI had infinite storage. However, if this assumption was not made, the algorithm design would be completely different and focusing more on avoiding

[9]function internal_work_orders() in total_order.py

bottlenecks in the flow rather than focusing on minimizing the vacancy time of the machines in the facilities.

In the process of distributing the total order, packaging type was not taken into account at any stage and was assigned at the end facility wise. However, the results can be improved by better assigning the packaging type. The packaging type 'Bag' takes more time to process. The entire order could have been distributed with the packaging type 'Box', and from the simulation results, the facilities that took the least amount of time could have been assigned more 'Bag' packages than the others. This would help better distribute the time and would result in a lower total time.

Moreover, when creating internal work orders, instead of taking each combination as an order, the quantity of each combination could be broken done to smaller quantities and rebuilt by having a certain ratio of the time approximation between consecutive orders. The results would improve by optimizing added parameters to find the best ratios.

In conclusion, this algorithm does an effective job of creating internal work orders for the firm. The algorithm can be used with varied number of equipment and their rates at each facility. By manipulating the numbers and observing the performance of the production line using the simulation, the firm can make investment decisions to make changes at facilities and improve its production rate.