THE FORGOTTEN REFRIGERATOR @District Market Alder
INCREASING THE EXPOSURE OF PREPARED FOOD

ZIHAN CHEN, YOUNG Z, KEVIN HUANG, JIAPENG NI

Abstract. This paper addresses the challenge of optimizing the layout of refrigerated items at *District Market Alder* (DM) to mitigate the impact of long queues during peak hours. The primary objective is to increase the quantity of high-demand refrigerated items by improving their accessibility through strategic rearrangement within the target refrigerators. By interacting with our community partner, *Kiel Turner*, manager of program operations, we modeled and solved the problem optimally using standard mathematical programming solvers. The results demonstrate a possibility of how a systematic approach to item rearrangement within refrigerators can mitigate the negative impact of long queues, ensuring that high-demand items are more easily accessible and thereby increasing overall sold net quantity. We hope this methodology can be extended to other retail environments to enhance the accessibility and sales of popular items in the future.

1. Introduction

District Market Alder (DM)[4] stands as the quintessential neighborhood grocery store, perfectly suited to meet the needs of the bustling university community. *DM* offers a wide variety of fresh products, ensuring top-notch quality and flavor for its customers. Its full-service deli provides chef-prepared sides, salads, and hot meals, ideal for those seeking a quick and nutritious eating option. Additionally, *DM* offers an extensive range of household items, health and beauty products, frozen foods, and a large selection of snacks and cold drinks. Dedicated to providing convenience, exceptional quality, and a pleasurable shopping experience, *DM* serves as a vital resource for all its patrons. Our community partner, *Kiel Turner*, the General Manager of *District Market Alder* and *Husky Grind Coffee*, told us that running the *DM* dedicated to providing excellent service to students is not an easy task. There are always inevitable challenges encountered during operation.

Our project aims to address the inefficiencies observed during peak hours at *DM*, particularly focusing on the long queues composed of people waiting to buy the *Freshly Prepared Food* (FPF), as shown in Figure 1 below. These queues often obstruct customers from accessing popular items in the Refrigerator section during the lunch and dinner rush, potentially negatively impacting sales. For example, the frozen prepared food section in Refrigerator 1 (R_1) has consistently been one of the most popular zones at DM. However, due to the layout in Figure 1, R_1 is frequently blocked by the line of people queuing at the FPF section.

The constant stream of people queuing during *peak hours*[1] often obstructs the flow of traffic to R_1, presenting one of the most pressing challenges for our community partner. The primary issue we aim to assist DM in resolving is determining the placement of each item in the six refrigerators to increase the quantity of items sold per week by reducing the *probability*[2] that customers will be blocked by the line.

Date: June 5, 2024.

[1]We define *peak hours* as 11:30-13:00 and 16:30-18:00 according to the *Transaction* Data provided by our community partner, which shows the highest customer traffic during these times.

[2]We will provide a detailed explanation of the *probability* in Section 3.1.

Therefore, we propose a discrete model that strategically swaps items between different refrigerators to ensure that high-demand items are more easily accessible, while less popular items are moved to obstructed areas, as illustrated by R_1 in Figure 1.

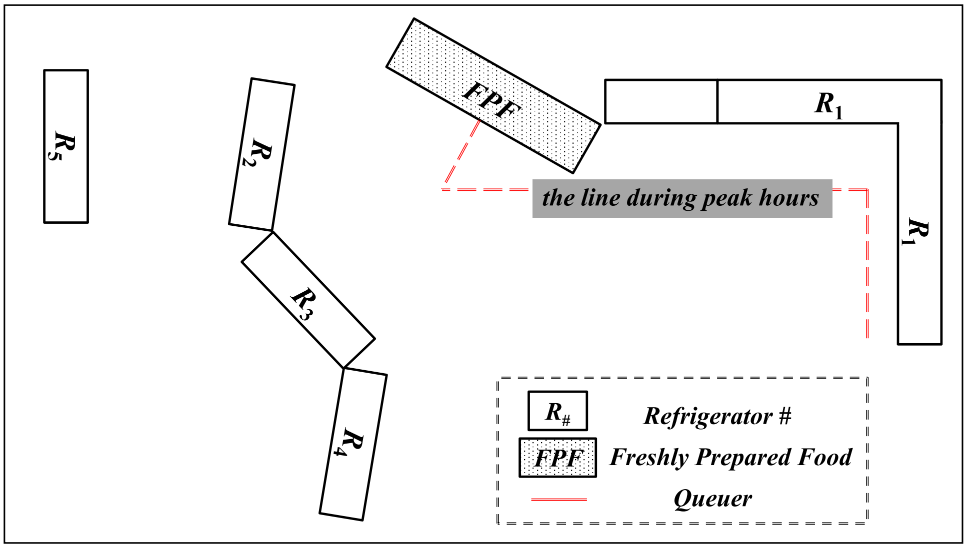

Figure 1: Target Refrigerators Layout @DM

2. Data Collection

Our community partner generously provided necessary data to support the development of our project, including weekly *Average Transactions Listed by 30 Min* and *Sales Report Summary Grouped by Categories, Groups, and Menu Items.* With these data, we could preliminarily understand the assortment of items for sale and their popularity.

Since our focus is on refrigerators, extracting these data was necessary but posed some challenges. Initially, we categorized the items in the table into three groups: those that definitely won't be stored in the refrigerators (such as bubble gum, condiments, and puffed food), those that might be stored in the refrigerators (like some desserts, snacks, and boxed fruits), and those that definitely will be stored in the refrigerators (such as dairy products and frozen prepared foods). This classification immediately reduced our dataset from 2300 rows to 600 rows. Our next step was to visit DM to physically confirm what exactly was stored in our *target refrigerators*[3]. To clearly illustrate our data processing steps at this stage, we have created the flowchart shown in Figure 2 below. Finally, we identified 176 *distinct items*[4] that were in 5 refrigerators.

Besides figuring out what is stored in each refrigerator, we needed to consider the dimensions of each shelf of the refrigerator to make sure our exchange of items is feasible. We must ensure that the dimensions of the items in our proposed strategy fit within the dimensions of each shelf of the refrigerators; otherwise, our exchange strategy would be impractical. We have a total of five refrigerators. We measured the length, width, and height of each shelf within these refrigerators. We also measured the size of each item's packaging box, keeping the measurement error within 1cm to ensure the accuracy of our measurements as much as possible. These data are indispensable for our project.

[3]Target refrigerators are the five refrigerators shown in Figure 1: R_1, R_2, R_3, R_4, and R_5.

[4]Even if some of them are different flavors of the same brand, we still consider them as distinct items.

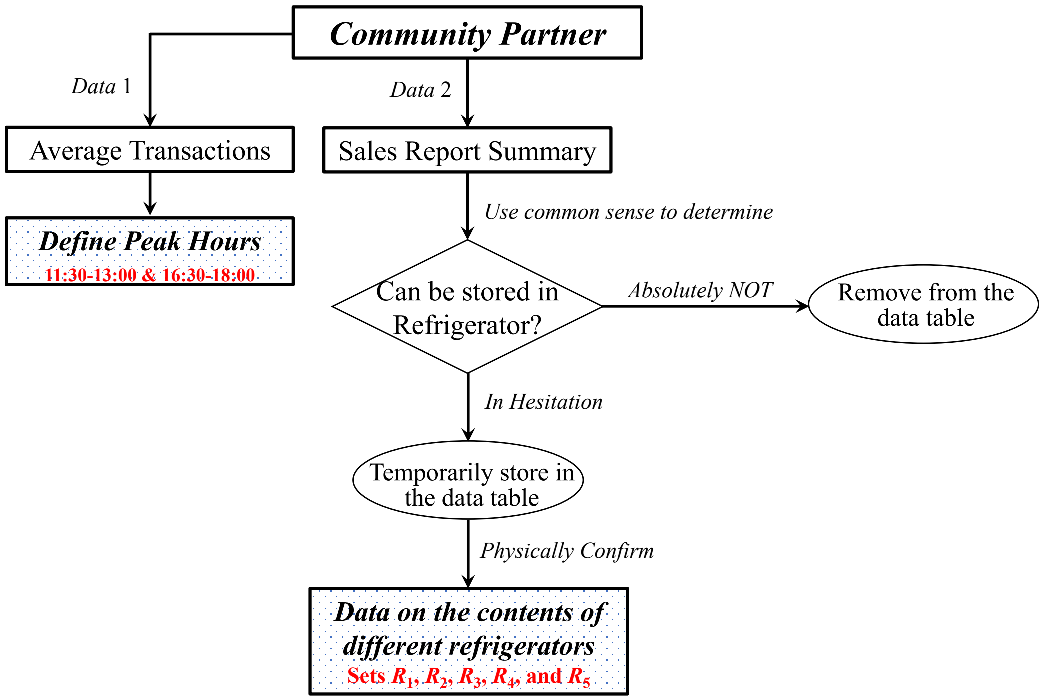

Figure 2: Data Processing

3. Mathematical Modeling

During our modeling process, we will use i to denote the i-th item. Based on the data processing in Figure 2, we can divide the 176 items among the different refrigerators R_1, R_2, R_3, R_4, and R_5 as follows:

$$
\begin{aligned}
R_1 &= \{i \in \mathbb{Z} \mid 1 \leq i \leq 49\},\\
R_2 &= \{i \in \mathbb{Z} \mid 50 \leq i \leq 72\},\\
R_3 &= \{i \in \mathbb{Z} \mid 73 \leq i \leq 104\},\\
R_4 &= \{i \in \mathbb{Z} \mid 105 \leq i \leq 116\},\\
R_5 &= \{i \in \mathbb{Z} \mid 117 \leq i \leq 176\}.
\end{aligned}
$$

We use 5 different sets R_1, R_2, R_3, R_4, and R_5 to include their respective items. For example, $R_2 = \{i \in \mathbb{Z} \mid 50 \leq i \leq 72\}$ implies that the 50^{th} to the 72^{nd} items are in set R_2, meaning they are in Refrigerator 2. To determine the popularity of each item i, we ranked all 176 items across the 5 refrigerators, denoting the rank of the i-th item as t_i, where $i \in \{1, 2, \ldots, 176\}$. These rankings were based on our existing weekly *Sales Reported Summary*, and t_i was generated using the `RANK` function in Microsoft Excel. Additionally, we defined the corresponding *average rank* for each refrigerator R_1, R_2, R_3, R_4, and R_5 as $\bar{T}_1, \bar{T}_2, \bar{T}_3, \bar{T}_4$, and $\bar{T}_5$. These average ranks were calculated using the following formulas (inspired by 5.1 in [1]):

$$
\begin{aligned}
\bar{T}_1 &= \frac{\sum_{i=1}^{49} t_i}{|R_1|} = \frac{\sum_{i=1}^{49} t_i}{49},\\
\bar{T}_2 &= \frac{\sum_{i=50}^{72} t_i}{|R_2|} = \frac{\sum_{i=50}^{72} t_i}{23},\\
\bar{T}_3 &= \frac{\sum_{i=73}^{104} t_i}{|R_3|} = \frac{\sum_{i=73}^{104} t_i}{32},
\end{aligned}
$$

$$\bar{T}_4 = \frac{\sum_{i=105}^{116} t_i}{|R_4|} = \frac{\sum_{i=105}^{116} t_i}{12},$$
$$\bar{T}_5 = \frac{\sum_{i=117}^{176} t_i}{|R_5|} = \frac{\sum_{i=117}^{176} t_i}{60}.$$

The notation $|R_1|$ above indicates the *cardinality*[5] of the set R_1, which is the number of elements in R_1, i.e., the number of items in Refrigerator 1, specifically 49. We obtain the *average rank* of Refrigerator 1, $\bar{T}_1$, by dividing the sum of the ranks of all items in Refrigerator 1 ($\sum_{i=1}^{49} t_i$) by the total number of items in Refrigerator 1, which is 49. We use the same method to calculate for $\bar{T}_2, \bar{T}_3, \bar{T}_4$ and $\bar{T}_5$. Therefore we got the *average rank* $\bar{T}$ for each refrigerator below:

$$\bar{T}_1 = 54.8,\ \bar{T}_2 = 133.7,\ \bar{T}_3 = 58.8,\ \bar{T}_4 = 85,\ \bar{T}_5 = 115.3.$$

According to the ranking criteria, the smaller the value of $\bar{T}$ is, the more popular all items in this refrigerator are. So according to $\bar{T}$, we rank popularity of 5 refrigerators as below:

$$\text{rank}(R_1) > \text{rank}(R_3) > \text{rank}(R_4) > \text{rank}(R_5) > \text{rank}(R_2).$$

Here, $\text{rank}(R_1) > \text{rank}(R_3)$ implies items in R_1 are more popular than items in R_3. From this perspective, the concerns of our community partner are not unfounded: the most popular items in R_1 are blocked during peak hours, which means that people are indeed likely to buy fewer of these popular items due to this obstruction. Items in R_1 are more favored by customers and indeed require greater visibility. On the contrary, items in R_2 and R_5 are not blocked by queues during peak hours, but their average ranks $\bar{T}$ are not as ideal. Therefore, we have decided that swapping items among R_1, R_2, and R_5 would certainly be a good choice. In the following steps, we will consider focusing solely on the items in R_1, R_2, and R_5.

3.1. **Probability Model.**

In our quest to understand consumer spending habits, we have developed our *Probability Model*. Firstly, we assume that each person entering the *District Market* will purchase one item and that the total number of items placed in Refrigerators R_1, R_2, R_3, R_4, and R_5 at *District Market* each week is consistent. We aim to calculate the probability of each person entering the *District Market* buying a specific item i based on weekly data. We define two *events*[6] as follows:

(Event 1) K_i = the person entering the market buys item i.

Here, $i \in \{1, 2, \cdots, 176\}$ represents all items in the refrigerators. For example, K_{10} would denote the event that a person entering the market buys the item 10.

(Event 2) B_a = the person will buy items in blocked R_1 in the a^{th} *configuration.*

We use the term "configuration" to describe the different placement strategies we have devised for the items. When $a = 0$, it refers to the current placement of items at *DM*, which is the initial configuration where no changes have been made. The letter a in B_a represents which configuration we are referring to. In this paper, we have 3 configurations (i.e., $a \in \{0, 1, 2\}$). We will provide a more comprehensive explanation of the different configurations in Section 3.2 Swapping Strategies.

Let q_i represent the net quantity of item i sold in a week, and Q be the total net quantity of items sold in a week. We aim to calculate the probability of a person entering the market buying item i. The probability of the event K_i, denoted as $P(K_i)$, is calculated using the following equation:[7]

$$P(K_i) = \frac{q_i}{Q}.$$

[5]The cardinality of R_1 may alternatively be denoted by $\text{card}(R_1)$ or $n(R_1)$.

[6]In mathematics, an "event" typically refers to a possible outcome or situation that can occur.

[7]We refer to the calculation of probability in Section 12.3, *Basic Rules of Probability*, of [7].

After we calculate $P(K_i)$ for each item i in 5 refrigerators, we need to sum the probability of people entering the *District Market* and buying item i in Refrigerator 1 (R_1), that is, $\sum_{i \in R_1} P(K_i)$, under different configurations(a). This gives us the probability that *people will buy items in blocked Refrigerator* $1(R_1)$ *in* a^{th} *configuration*, which we denoted as $P(B_a)$. We are particularly interested in the items in R_1 because R_1 is the only refrigerator affected by this issue. Following the instructions above, we have the equation:

$$P(B_a) = \sum_{i \in R_1} P(K_i).$$

By calculating the probability that people will buy the item in R_1 in different configurations $a \in \{0, 1, 2\}$, we can compare the probability $P(B_a)$ that people will choose to buy the item in R_1 in different configurations and finally get the optimal configuration a, whose $P(B_a)$ is minimized and the configuration is reasonable. Therefore, we aim to minimize the probability $P(B_a)$ as much as possible by altering our configuration. However, it's important to note that we must also consider additional constraints related to the length, width, and height of each item and each shelf in different refrigerators, as discussed in Section 3.3 Adding Constraints.

3.2. **Swapping Strategies.**

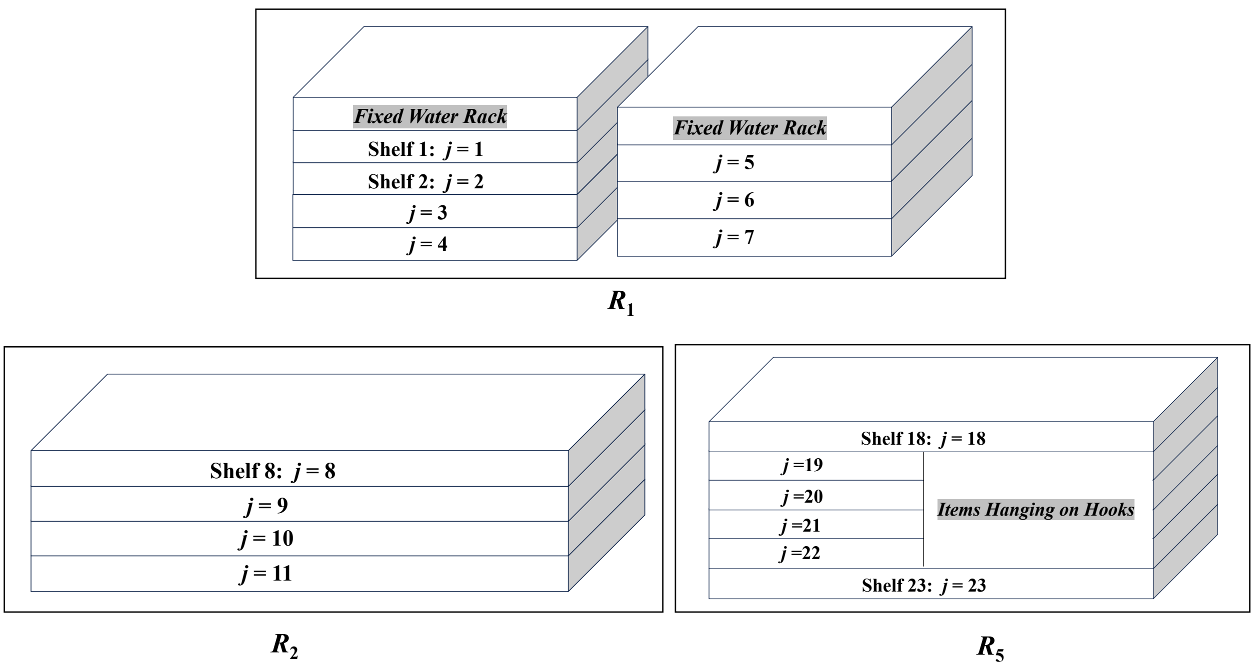

Figure 3: Shelves in R_1, R_2, and R_5

In this part, we are going to swap items to increase the future net quantity of items in R_1, R_2, and R_5. We included the structure of these three refrigerators in Figure 3 above. As can be seen from Figure 3, there are specific items — *"fixed items"* — that we are unable to swap including beverages stored on a *Fixed Water Rack* and *Items Hanging on Hooks* in R_1 and R_5 separately. After removing the shelves containing the *"fixed items"*, we only need to consider the available shelves j in R_1, R_2, and R_5, where $j \in \{1, 2, 3, 4, 5, 6, 7\} \cup \{8, 9, 10, 11\} \cup \{18, 19, 20, 21, 22, 23\}$ according to Figure 3. We omitted the 12^{nd} to the 17^{th} shelves in R_3 and R_4 because we do not consider swapping items in these two refrigerators. Next, we are going to introduce the two swapping strategies.

3.2.1. *Strategy 1: Swap based on rank* (Configuration $a = 1$).

After removing the *fixed items* from our exchange list, we have exactly 34 items available for exchanging in R_1, which we consider as 34 slots. In *Strategy 1*, we aim to fill the 34 slots in R_1 with the 34 lowest-ranked items based on their rankings t_i across R_1, R_2, and R_5. As an example, Table 1 below shows the 14 lowest-ranked items. We propose placing the lowest-ranked 34 items, including those in Table 1, into R_1. In other words, in Strategy 1, we keep item 12 still in R_1. We propose moving the items that were originally in R_2 and R_5 (in our example, $i = 57, 126, 172, 176$) to R_1. Additionally, if the rankings of items(t_i) in R_2 and R_5 are not among the lowest 34, we will not move these items, keeping them in their original refrigerators. Considering the items in R_1 that are not among the lowest 34, we will move them to the remaining slots in R_2 and R_5 by guaranteeing the same brand items are placed together as much as possible.

Item i	Item Menu	Initial Refrigerator $R_{\#}$	Rank t_i
57	Neato Bur Thai Delight	2	163
⋮	⋮	⋮	⋮
126	Field Roast Chao Cheese Original	5	167
⋮	⋮	⋮	⋮
12	Egg Hard Boiled Individual	1	174
172	Field Roast Lentil Sage Deli	5	175
176	Uptons Natural Seitan Bacon	5	176

Table 1: The 14 Lowest-Ranked items

This process will result in an updated R_1. This approach corresponds to event B_1 in our first configuration, where $a = 1$.

3.2.2. *Strategy 2: Swap based on brand* (Configuration $a = 2$).

In Strategy 2, we aim to fill the 34 *non-fixed* slots of R_1(as we discussed before). Initially, we will categorize the items into several groups based on the *brand*, then select those groups with relatively low rankings to place in R_1. Set an example table below, it is evident that items 169, 170, 171, and 172 are of the same brand "*Field Roast*". Moreover, we observe that items in this brand generally have relatively low rankings t_i since the lowest ranking value is 176. We define such items of the same brand with generally low rankings in Table 2 as *group* in our *Strategy 2*. Our strategy will also be conducted on a group-by-group basis.

Item i	Menu Item	Rank t_i
169	Field Roast Brkfst Mple Sausage	171
170	Field Roast Frankforter Sausage	172
171	Field Roast Fruffalo Wing	173
172	Field Roast Lentil Sage Deli	175

Table 2: Same Brand Items

If the items in R_2 and R_5 are not part of the relatively low-ranking groups of the same brand, we will not move these items and they will remain in R_2 and R_5. Similarly, we will fill the remaining slots in R_2 and R_5 with items from R_1 that are not part of the relatively low-ranking *groups* while ensuring the same brand items are placed together as much as possible. This approach corresponds to event B_2 in our second configuration, where $a = 2$.

3.3. **Adding Constraints.**

As discussed in Section 3.1 Probability Model, when considering reducing the value of $P(B_a)$ by different configurations a, we need to consider the constraints that the sum of the length, width, and height of items on the shelf should not exceed the length, width, and height of the shelf itself. We will define our variables and set up our inequalities below.

To facilitate our programming operations, we use a_{ij}, b_{ij}, c_{ij} as our variables, where a_{ij}, b_{ij}, c_{ij} should be positive integers, $i \in \{1, 2, \cdots, 176\}$ denotes the item i, and $j \in \{1, 2 \cdots, 23\}$ denotes the shelf j. Let a_{ij} be the number of *columns* that we are capable of placing the i^{th} items on the j^{th} shelf. As shown in Figure 4(a), we will arrange item i in 4 *columns* on shelf j, so here $a_{ij} = 4$. Likewise, we use b_{ij} to represent the number of *rows* that we are capable of placing the i^{th} items on the j^{th} shelf. Also can be seen from Figure 4(b), $b_{ij} = 3$. c_{ij} is the number of *stacks* that we are capable of placing the i^{th} items on the j^{th} shelf vertically. When items are arranged in the form shown in Figure 4(c), we denote that $c_{ij} = 2$.

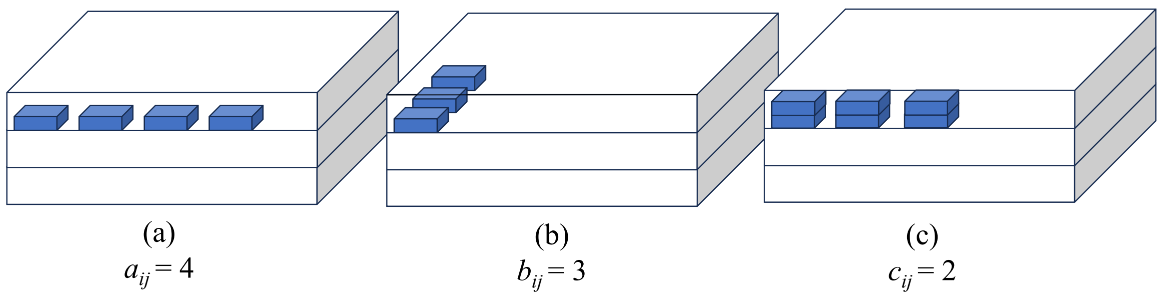

Figure 4: Different Arrangement Methods for Items[8]

Most of the literature we have reviewed (e.g., [3][8]) focuses more on a method aimed at maximizing spatial utilization for the placement of large items. These large items are often placed in a crisscross manner in the space. However, this is not ideal for the placement of *DM* items. We must consider placing similar items together and arranging them in an orderly manner. Nevertheless, the consideration of length, width, and height constraints in the aforementioned literature is something we can reference.

Considering that our item arrangement is constrained by the dimensions of each shelf in refrigerators, including length, width, and height, our configuration will be subject to the following constraints:

$$(1) \qquad \sum_{i \in R_1 \cup R_2 \cup R_5} l_i \cdot a_{ij} \leq L_j \qquad \forall j \in \{1, 2, \cdots, 7\} \cup \{8, \cdots 11\} \cup \{18, \cdots 23\}$$

Where l_i is the length of item i, and L_j is the length of the j^{th} shelf.

$$(2) \qquad w_i \cdot b_{ij} \leq W_j \qquad \forall i \in R_1 \cup R_2 \cup R_5,\ \forall j \in \{1, 2, \cdots, 7\} \cup \{8 \cdots 11\} \cup \{18 \cdots 23\}$$

Where w_i is the width of item i, and W_j is the width of the j^{th} shelf.

$$(3) \qquad h_i \cdot c_{ij} \leq H_j \qquad \forall i \in R_1 \cup R_2 \cup R_5,\ \forall j \in \{1, 2, \cdots, 7\} \cup \{8 \cdots 11\} \cup \{18 \cdots 23\}$$

Where h_i is the height of item i, and H_j is the height of j^{th} shelf. As can be seen from Figure 5 and Figure 6 below, we can easily recognize the l_i, w_i, h_i, and L_j, W_j, H_j.

[8]Note here, even for the same Refrigerator, the length, width, and height of each shelf may vary. For the sake of drawing and easier understanding, we draw the length, width, and height of each shelf as the same in Figure 4.

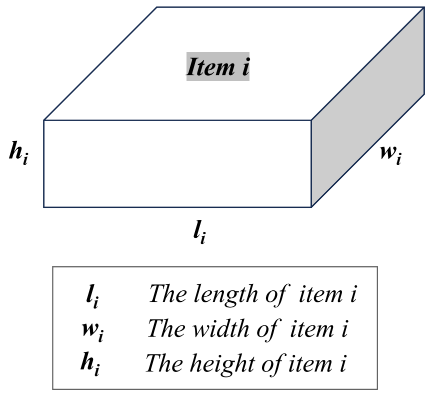

Figure 5:
Definition of Item Length, Width, and Height

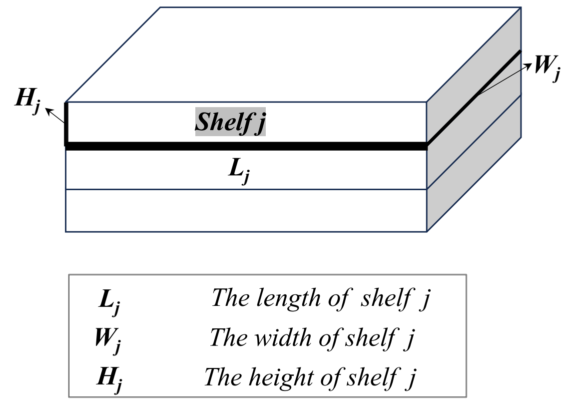

Figure 6:
Definition of Shelf Length, Width, and Height

4. Compiling

4.1. **Different Swapping Strategies.**

4.1.1. *Strategy 1.* Based on our Strategy 1, we can generate Table 4-1, Table 4-2, and Table 4-3, which respectively describe the items placed in R_1, R_2, and R_5 when the configuration is set to $a = 1$.

Table 4-1: R_1's Menu Item (under configuration $a = 1$)

Item i	Menu Item	Initial Refrigerator $R_{\#}$
3	Kraft Big Cheese Bar Colby Jack	1
⋮	⋮	⋮
51	Pennys Salsa Mild 16oz	2
⋮	⋮	⋮
117	BelGioioso Asiago Wedge	5
⋮	⋮	⋮

Table 4-2: R_2's Menu Item (under configuration $a = 1$)

Item i	Menu Item	Initial Refrigerator $R_{\#}$
23	DM2Go Fresh Cut Fruit	1
⋮	⋮	⋮
50	Pennys Salsa Medium 16oz	2
⋮	⋮	⋮

Table 4-3: R_5's Menu Item (under configuration $a = 1$)

Item i	Menu Item	Initial Refrigerator $R_{\#}$
4	Wilcox HB Egg 2Pk Cage Free	1
⋮	⋮	⋮
118	BelGioioso Mascarpone Belgioioso	5
⋮	⋮	⋮

The complete Table 4-1, Table 4-2, and Table 4-3 are attached in the Appendix 8.1.2. By examining the complete tables, we were surprised to find that, following *Strategy 1*, which involves arranging items strictly according to their rank, the items still tend to be grouped by brand.

4.1.2. *Strategy 2.* Based on our *Strategy 2*, we can generate the following Table 5-1, Table 5-2, and Table 5-3, which respectively describe the items placed in R_1, R_2, and R_5 when the configuration is set to $a = 2$.

Table 5-1: R_1's Menu Item (under configuration $a = 2$)

Item i	Menu Item	Initial Refrigerator $R_{\#}$
50	Pennys Salsa Medium 16oz	2
$\vdots$	$\vdots$	$\vdots$
117	BelGioioso Asiago Wedge	5
$\vdots$	$\vdots$	$\vdots$

Table 5-2: R_2's Menu Item (under configuration $a = 2$)

Item i	Menu Item	Initial Refrigerator $R_{\#}$
4	Wilcox HB Egg 2Pk Cage Free	1
$\vdots$	$\vdots$	$\vdots$
59	Marukome Miso Organic 13.2oz	2
$\vdots$	$\vdots$	$\vdots$

Table 5-3: R_5's Menu Item (under configuration $a = 2$)

Item i	Menu Item	Initial Refrigerator $R_{\#}$
3	Kraft Big Cheese Bar Colby Jack	1
$\vdots$	$\vdots$	$\vdots$
127	Galbani Mozzarella Loaf 16oz	5
$\vdots$	$\vdots$	$\vdots$

The complete Table 5-1, Table 5-2, and Table 5-3 are attached in the Appendix 8.1.3 as well. From these three tables, we can observe that the *Strategy 2* maximizes the grouping of similar items in the same refrigerator, which is our ideal scenario.

4.2. **Probability of different strategies.**

4.2.1. *Probability of initial placement: $P(B_0)$.*

We define the current item placement method at *DM* as the *initial placement* as shown in Appendix 8.1.1. We can calculate $P(B_0)$, the probability that people will buy the item in the Refrigerator $1(R_1)$ blocked by the line based on the probability model in Section 3.1. We calculate the sum of the sold net quantity of items in the initial R_1 (configuration $a = 0$) as 3534, and the total sold net quantity of items in all refrigerators is 6611 in Data Collection process. Therefore:

$$P(B_0) = \sum_{i \in R_1} P(J_i) = \sum_{i \in R_1} \frac{q_i}{Q} = \frac{\sum_{i \in R_1} q_i}{Q} = \frac{3534}{6611} = 0.507336258.$$

The situation does not seem ideal, as half of the customers want to buy items in Refrigerator 1 that are blocked by the queue.

4.2.2. *Probability of Strategy 1: $P(B_1)$.*

According to *Strategy 1* in 4.1.1, we will fill the 34 slots of R_1 with the 34 lowest-ranked items based on their rankings. Then, we compute the sum of the sold net quantity of items in the *updated* R_1 as 148 in the configuration $a = 1$. The total sold net quantity of items in refrigerators is still 6611. Then we calculate the $P(B_1)$ as :

$$P(B_1) = \frac{148}{6611} = 0.0223869.$$

It looks quite promising, indicating that our strategy has been very effective. By applying our *Strategy 1*, we've found that only $0.022(\approx 2\%)$ of people will buy items from blocked Refrigerator 1, meaning that the probability of people being obstructed when trying to buy items from Refrigerator 1 is greatly reduced.

4.2.3. *Probability of Strategy 2: $P(B_2)$.*

According to *Strategy 2* in 4.1.2, we will fill the 34 slots of R_1 with items in relatively low-ranking groups of the same brand. Then we found out the 34 items in this strategy and we computed the sum of the sold net quantity of items in R_1 as 394 in the configuration $a = 2$. Therefore, $P(B_2)$ can be expressed as:

$$P(B_2) = \frac{394}{6611} = 0.05959764.$$

Strategy 2 also looks good, but its effect is not as pronounced as *Strategy 1*, with still $0.059(\approx 6\%)$ of people purchasing items blocked in Refrigerator 1.

4.3. **Items Placement.**

Based on the current analysis, it appears that *Strategy 1* is the most effective in reducing $P(B_a)$. However, we must verify the feasibility of *Strategy 1*. We need to ensure that the *dimensions* of the items proposed for exchange are compatible with the *dimensions* of the corresponding shelves, as discussed in Section 3.3. Certainly, our goal goes beyond simply telling our community partner R_1, R_2, and R_5 what each refrigerator should contain; We also intend to present our specific plan for item placement to the manager. If we can develop a plan for item placement to present to the manager, it serves as an alternative way to verify that our configuration passes the feasibility test. This is a crucial part of the modeling process[1].

Thus, we consider Section 3.3 Adding Constraints, designed to optimally arrange items on shelves of varying lengths. In our first step, we intend to maximize space utilization by adjusting the number of columns each item can occupy in each shelf. In the process of organizing items, we adhere to a crucial principle of *fairness*, ensuring that every item on each shelf gets fair exposure by arranging them to occupy an equal number of columns as much as possible. Ideally, similar items should be grouped, mainly considering consumer habits. With our programming tool, PYTHON, we can easily achieve our objectives.

During the coding process, we group all items within each refrigerator according to the available shelves[9]. Set an example in Figure 7-1, if we try to distribute 34 items across 7 shelves, we initially set five items per shelf, with the last shelf holding four items[10]. We start by placing one column of each item and increase the number of columns if the length of the shelf permits. For instance, if we have different items A, B, C, D, E on the first shelf as shown in Figure 7-1, the code first checks if placing $A|B|C|D|E$ on the first shelf is feasible.

[9] As discussed before, available shelves $j \in \{1, 2, 3, 4, 5, 6, 7\} \cup \{8, 9, 10, 11\} \cup \{18, 19, 20, 21, 22, 23\}$.

[10] The main purpose of this arrangement is to ensure that every item on each shelf gets fair exposure.

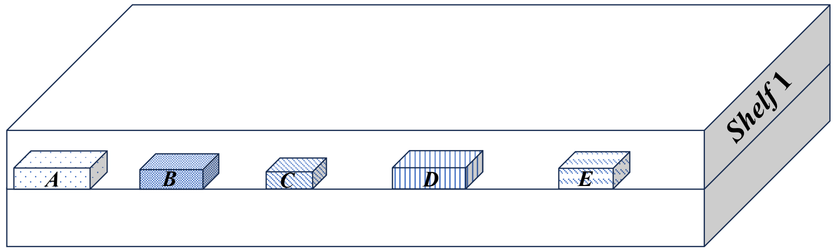

Figure 7-1: First Step of (Column) Placement

If placing $A|B|C|D|E$ on the first shelf is feasible, then check the feasibility of placing $AA|BB|CC$ $|DD|EE$ on the first shelf, as shown in Figure 7-2.

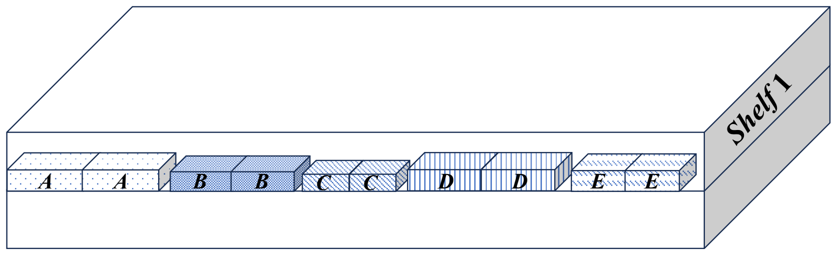

Figure 7-2: Second Step of (Column) Placement

Then we keep checking. If the layout of $AAA|BBB|CCC|DDD|EEE$ exceeds the length of the first shelf, we must pause, as we have reached the limit of averaging item arrangement. That is to say, we cannot place three columns of each item on the first shelf. We then consider if the remaining space can be effectively used by observing whether it can accommodate some of the A, B, C, D, and E items. If possible, this would be our ideal scenario, and our code would output the placement for maximizing length usage with the optimal item arrangement. Suppose we can accommodate one more item C as shown in Figure 7-3, thus, our optimal item arrangement for shelf 1 should be $a_{A1} = 2, a_{B1} = 2, a_{C1} = 3, a_{D1} = 2$, and $a_{E1} = 2$. Restating, $a_{A1} = 2$ means we suggest placing item A on shelf 1 in 2 columns, and $a_{C1} = 3$ means we suggest placing item C on shelf 1 in 3 columns. Translate into actionable steps, which means placing two columns of item A, two columns of item B, three columns of item C, two columns of item D, and two columns of item E on shelf 1.

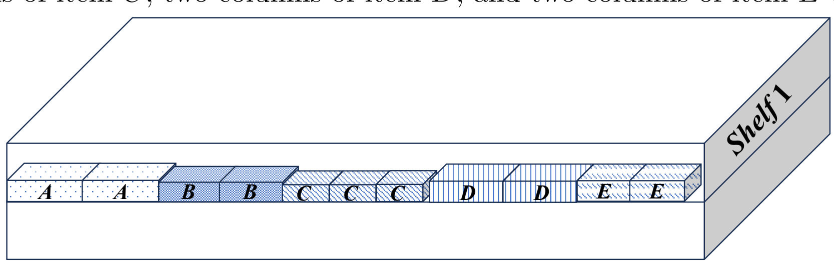

Figure 7-3: Third Step of (Column) Placement

This is our main strategy, but in the actual programming process, we considered that even for a refrigerator, the length of each shelf would be different. Therefore, we devised a *fairer* item arrangement strategy based on the length's proportions of each shelf (can be found in Appendix 8.2.1). For example, if we have 5 items to be allocated on two shelves with lengths of $120cm$ and $180cm$ respectively, we calculate the ratio of these two shelves as 2:3. Therefore, we propose placing 2 distinct items on the $120cm$ shelf and 3 distinct items on the $180cm$ shelf.

After considering how many columns each item can have on each shelf, we then consider how many rows each item can have within each shelf. In our notation in Section 3.3, we use b_{ij} to represent the number of rows each item i can have on shelf j. We only need to use the formula:

$$b_{ij} \le \frac{W_j}{w_j},\ b_{ij} \in \mathbb{Z}^+.$$

Here $b_{ij} \in \mathbb{Z}^+$ implies b_{ij} can only be a positive integer. The code used to calculate the specific arrangement of items is provided in Appendix 8.2.2 – 8.2.4.

Regarding the number of stacks, we can precisely calculate c_{ij}, which represents the number of stacks item i that can be placed on shelf j, but here we just assume each item's c_{ij} will be 1 because we are not clear about *DM*'s placement strategy regarding the number of stacks. Some items' quality may be undermined if we place too many stacks together. We also need to consider external factors like *airflow*. We believe that *DM* has expertise in displaying items, so we have proposed only one feasible way of arranging items in Appendix 8.2.5.

5. Conclusion

By comparing $P(B_0)$, $P(B_1)$ and $P(B_2)$, $P(B_1) < P(B_2) \ll P(B_0)$, we conclude that our strategy is indeed effective, and B_1 is more effective than B_2. We were surprised to find that in configuration $a = 1$, even when we swap items according to their ranks, items of the same brand, and even of the same type, still cluster together. This phenomenon was not surprising to our community partner. He told us that it is closely related to *consumer psychology*. Past business experiences have shown him that if consumers like a particular product from a brand, they are more likely to buy other products from the same brand. Conversely, if they are indifferent to a product, they are less likely to purchase any products from that brand. [11]

Therefore, we can summarize that $a = 1$ is the optimal configuration which minimizes the probability of the event that people will be blocked and keeps as many items of the same category as possible in the refrigerator. After that, we checked that the optimal configuration is feasible since we developed a reasonable item arrangement plan for configuration $a = 1$ in Appendix 8.2.5.

6. Future Implications

District Market undergoes a large-scale overhaul every summer vacation. Our community partner informed us that this is the best time to try our *Strategy*. We are very eager to see the sales performance of the items after using our suggested configuration. We believe it will certainly reduce the probability of customers being obstructed, but we are more interested in using real-life data they will provide us in the future to verify the reliability of our model.

We also look forward to continuing to improve the quality of our models in the future. Currently, our probability predictions are primarily based on weekly sales data and do not take other factors into account, such as the introduction of new products into the District Market's refrigerator section. Additionally, if we have the opportunity in the future, we hope to consider more factors in deciding the placement in Section 4.3, such as determining the number of items to be placed based on their ranking. We hope this is not the end for us. We are more committed to creating a project that can provide item placement strategies for other markets at *UW* with similar functions.

7. Acknowledgement

First, we would like to thank our community partner *Kiel Turner* for his support. Without him, this project would not have been possible. We appreciate his trust and his generous provision of ample data, which gave us the confidence to complete this project. We also extend our heartfelt gratitude to Prof. *Sara Billey* for her unwavering guidance, countless feedback sessions, and for witnessing the growth of this project with us.

Additionally, we are grateful to our TA *Tony Zeng* for his feedback on our first draft, as well as to the *Meany Hall* and *Center Table* groups for their crucial feedback during the peer review phase. Most importantly, we want to thank everyone who provided feedback or encouragement during the poster session. Our deepest gratitude goes to all the aforementioned individuals, whose love and trust made this project better. Your feedback, much like this *Refrigerator* in our project, will forever remain unforgotten in our hearts.

[11] We have further discussions on the *Economic Perspective on Item Placement* in Appendix 8.1.4.

References

[1] Sara Billey et al. *Discrete Mathematical Modeling: Math 381 Course Notes*. University of Washington, 2010.

[2] DM Group. *Github Repository. 2024.* https://github.com/Youni2017/Math381-FinalProject.git. Accessed: 05/06/2024.

[3] Silvano Martello et al. "Algorithm 864: General and robot-packable variants of the three-dimensional bin packing problem". In: *ACM Trans. Math. Softw.* 33.1 (Mar. 2007), 7–es. ISSN: 0098-3500. DOI: 10.1145/1206040.1206047. URL: https://doi-org.offcampus.lib.washington.edu/10.1145/1206040.1206047.

[4] University of Washington News. *Groceries and more: District Market opens in Alder Hall*. 2012. URL: https://www.washington.edu/news/2012/09/12/groceries-and-more-district-market-opens-in-alder-hall/ (visited on 05/25/2024).

[5] Wikipedia contributors. *Attention economy — Wikipedia, The Free Encyclopedia*. [Online; accessed 5-June-2024]. 2024. URL: https://en.wikipedia.org/w/index.php?title=Attention_economy&oldid=1217490400.

[6] Wikipedia contributors. *Substitute good — Wikipedia, The Free Encyclopedia*. [Online; accessed 5-June-2024]. 2024. URL: https://en.wikipedia.org/w/index.php?title=Substitute_good&oldid=1221294872.

[7] W.L. Winston. *Operations Research: Applications and Algorithms*. Cengage Learning, 2022. ISBN: 9780357907818. URL: https://books.google.com/books?id=Y9NYEAAAQBAJ.

[8] S. Zhou et al. "Two-dimensional knapsack-block packing problem". In: *Applied Mathematical Modelling* 73 (2019), pp. 1–18. DOI: 10.1016/j.apm.2019.03.039.

8. Appendix

All the files can be found in [2].

8.1. Items in R_1, R_2, and R_5 under different strategies.

8.1.1. *Strategy 0.* As we mentioned before, $a = 0$ refers to the initial placement of items in refrigerators R_1, R_2, and R_5 at DM. The following Table 3 shows the original positions of the items.

Table 3: R_1, R_2, and R_5's Menu Item (under configuration $a = 0$)

Item i	Menu Item	Initial Refrigerator $R_{\#}$
3	Kraft Big Cheese Bar Colby Jack	1
4	Wilcox HB Egg 2Pk Cage Free	1
5	Columbus Snack Tray Italian Dry Salami	1
6	Hillshire SP Genoa Salame	1
7	Hillshire SP Hot Calabrese	1
8	Hillshire SP Ital Dry Salame	1
9	Sabra Hummus Garlic and Pretzel	1
10	Sabra Hummus Red Pepper	1
12	Egg Hard Boiled Individual	1
13	BBQ Turkey Meatballs	1
14	Chicken Potsticker Salad	1
15	Corn Salad	1
16	Pesto Tortellini	1
17	Quinoa Tabbolouleh	1
18	Roasted Broccoli Salad	1
19	Tropical Fruit Salad	1

20	Teriyaki Beef Meal	1
21	Teriyaki Chicken Meal	1
22	Vegan Chana Masala	1
23	DM2Go Fresh Cut Fruit	1
24	DM2Go Chia Pudding Strawberry	1
25	DM2Go Fresh Crudite	1
26	DM2Go Fresh Grapes	1
27	DM2Go Moto Hawaiian Fried Rice	1
28	DM2Go Chicken Kale Caesar Salad	1
29	DM2Go Farmers Market Salad	1
30	DM2Go Greek Salad	1
31	DM2Go Chicken Shawarma	1
32	DM2Go Ham Havarti Croissant Sandwich	1
33	DM2Go Mediterranean Chickpea Sandwich	1
34	DM2Go Peanut Butter and Jam Sandwich	1
35	DM2Go Turkey and Cheddar Sandwich	1
48	G2G Bar Choc Banana	1
49	G2G Bar Peanut Butter & Jelly	1
50	Pennys Salsa Medium 16oz	2
51	Pennys Salsa Mild 16oz	2
52	Neato Bur Albuquerque	2
53	Neato Bur Paradise	2
54	Neato Bur Seven Layer	2
55	Neato Bur South of Border	2
56	Neato Bur Spicy Potato	2
57	Neato Bur Thai Delight	2
58	Neato Hummus Chipotle	2
59	Marukome Miso Organic 13.2oz	2
60	Marukome Miso Organic LS 13.2oz	2
61	Columbus Pastrami Presliced 6oz	2
62	Olli Pepperoni Presliced 4oz	2
63	Columbus Charcuterie Tasting Board 4oz	2
65	Better Bean Hummus Cup 2oz	2
66	Divina Olives Blue Cheese Stuffed 4.6oz	2
67	Divina Olives Castelvetrano 4.6oz	2
68	Divina Olives Kalamata Pitted 4.2oz	2
69	GeradDom Wild Coho Smoked Salmon	2
70	SarbarGoMed Hummus	2
71	SarbarGoMed Hummus Rstd Garlic	2
72	SarbarGoMed Hummus Rstd Pep	2
117	BelGioioso Asiago Wedge	5
118	BelGioioso Mascarpone Belgioioso	5
119	BelGioioso Parmesan Shredded Cup	5
120	BelGioioso Parmesan Wedge	5
121	BelGioioso Romano Shredded Cup	5
122	BelGioioso Romano Wedge	5
123	BelGioioso Snacking Mozzarella	5
124	Best Yet Amer Singles 12oz	5
125	Best Yet Colby Jack Shred 8oz	5
126	Field Roast Chao Cheese Original	5

127	Galbani Mozzarella Loaf 16oz	5
128	Kraft Amer Singles 8oz	5
133	Laura Chenel Goat Plain Log	5
134	Laural Chenel Garlic & Chive Goat Log 4oz	5
135	Organic Creamery Feta Crumble Cup	5
153	Boursin Cheese Garlic Herb 5oz	5
154	Daiya Vegan Cream Cheeese Strawberry 8oz	5
155	Daiya Vegan Cream Cheese Plain 8oz	5
156	Daiya Vegan Sliced Cheese	5
157	Philadelphia Cream Cheese Low Fat	5
158	Philadelphia Cream Cheese Original	5
159	Beechers Mac and Cheese Gluten Free	5
160	Beechers Mac and Cheese Worlds Best	5
161	Buddig Ham 2oz	5
162	Oscar M Deli Roasted Turkey 9oz	5
163	Applegate Chicken Maple Sausag	5
164	Applegate Oven Roasted Chicken	5
165	Applegate Sliced Ham	5
166	Applegate Smoked Turkey Slices	5
167	Applegate Turkey Bacon	5
168	Field Roast Apple Sage Sausage	5
169	Field Roast Brkfst Mple Sausage	5
170	Field Roast Frankforter Sausage	5
171	Field Roast Fruffalo Wing	5
172	Field Roast Lentil Sage Deli	5
173	Field Roast Mex Chiptle Sausage	5
174	Field Roast Smked Tmto Deli Sliced	5
175	Uptons Natural Seitain Chorizo	5
176	Uptons Natural Seitan Bacon	5

8.1.2. *Strategy 1.* Here is the suggested placement of items in refrigerators R_1, R_2, and R_5 after applying strategy 1, i.e., for configuration $a = 1$.

Table 4-1: R_1's Menu Item (under configuration $a = 1$)

Item i	Menu Item	Initial Refrigerator $R_{\#}$
3	Kraft Big Cheese Bar Colby Jack	1
12	Egg Hard Boiled Individual	1
51	Pennys Salsa Mild 16oz	2
52	Neato Bur Albuquerque	2
53	Neato Bur Paradise	2
55	Neato Bur South of Border	2
56	Neato Bur Spicy Potato	2
57	Neato Bur Thai Delight	2
58	Neato Hummus Chipotle	2
59	Marukome Miso Organic 13.2oz	2
60	Marukome Miso Organic LS 13.2oz	2
61	Columbus Pastrami Presliced 6oz	2
63	Columbus Charcuterie Tasting Board 4oz	2
65	Better Bean Hummus Cup 2oz	2
66	Divina Olives Blue Cheese Stuffed 4.6oz	2

67	Divina Olives Castelvetrano 4.6oz	2
68	Divina Olives Kalamata Pitted 4.2oz	2
117	BelGioioso Asiago Wedge	5
122	BelGioioso Romano Wedge	5
126	Field Roast Chao Cheese Original	5
133	Laura Chenel Goat Plain Log	5
134	Laural Chenel Garlic & Chive Goat Log 4oz	5
154	Daiya Vegan Cream Cheeese Strawberry 8oz	5
155	Daiya Vegan Cream Cheese Plain 8oz	5
156	Daiya Vegan Sliced Cheese	5
168	Field Roast Apple Sage Sausage	5
169	Field Roast Brkfst Mple Sausage	5
170	Field Roast Frankforter Sausage	5
171	Field Roast Fruffalo Wing	5
172	Field Roast Lentil Sage Deli	5
173	Field Roast Mex Chiptle Sausage	5
174	Field Roast Smked Tmto Deli Sliced	5
175	Uptons Natural Seitain Chorizo	5
176	Uptons Natural Seitan Bacon	5

Table 4-2: R_2's Menu Item (under configuration $a = 1$)

Item i	Menu Item	Initial Refrigerator $R_{\#}$
23	DM2Go Fresh Cut Fruit	1
24	DM2Go Chia Pudding Strawberry	1
25	DM2Go Fresh Crudite	1
26	DM2Go Fresh Grapes	1
27	DM2Go Moto Hawaiian Fried Rice	1
28	DM2Go Chicken Kale Caesar Salad	1
29	DM2Go Farmers Market Salad	1
30	DM2Go Greek Salad	1
31	DM2Go Chicken Shawarma	1
32	DM2Go Ham Havarti Croissant Sandwich	1
33	DM2Go Mediterranean Chickpea Sandwich	1
34	DM2Go Peanut Butter and Jam Sandwich	1
35	DM2Go Turkey and Cheddar Sandwich	1
48	G2G Bar Choc Banana	1
49	G2G Bar Peanut Butter & Jelly	1
50	Pennys Salsa Medium 16oz	2
54	Neato Bur Seven Layer	2
62	Olli Pepperoni Presliced 4oz	2
69	GeradDom Wild Coho Smoked Salmon	2
70	SarbarGoMed Hummus	2
71	SarbarGoMed Hummus Rstd Garlic	2
72	SarbarGoMed Hummus Rstd Pep	2

Table 4-3: R_5's Menu Item (under configuration $a = 1$)

Item i	Menu Item	Initial Refrigerator $R_{\#}$

4	Wilcox HB Egg 2Pk Cage Free	1
5	Columbus Snack Tray Italian Dry Salami	1
6	Hillshire SP Genoa Salame	1
7	Hillshire SP Hot Calabrese	1
8	Hillshire SP Ital Dry Salame	1
9	Sabra Hummus Garlic and Pretzel	1
10	Sabra Hummus Red Pepper	1
13	BBQ Turkey Meatballs	1
14	Chicken Potsticker Salad	1
15	Corn Salad	1
16	Pesto Tortellini	1
17	Quinoa Tabbolouleh	1
18	Roasted Broccoli Salad	1
19	Tropical Fruit Salad	1
20	Teriyaki Beef Meal	1
21	Teriyaki Chicken Meal	1
22	Vegan Chana Masala	1
118	BelGioioso Mascarpone Belgioioso	5
119	BelGioioso Parmesan Shredded Cup	5
120	BelGioioso Parmesan Wedge	5
121	BelGioioso Romano Shredded Cup	5
123	BelGioioso Snacking Mozzarella	5
124	Best Yet Amer Singles 12oz	5
125	Best Yet Colby Jack Shred 8oz	5
127	Galbani Mozzarella Loaf 16oz	5
128	Kraft Amer Singles 8oz	5
135	Organic Creamery Feta Crumble Cup	5
153	Boursin Cheese Garlic Herb 5oz	5
157	Philadelphia Cream Cheese Low Fat	5
158	Philadelphia Cream Cheese Original	5
159	Beechers Mac and Cheese Gluten Free	5
160	Beechers Mac and Cheese Worlds Best	5
161	Buddig Ham 2oz	5
162	Oscar M Deli Roasted Turkey 9oz	5
163	Applegate Chicken Maple Sausag	5
164	Applegate Oven Roasted Chicken	5
165	Applegate Sliced Ham	5
166	Applegate Smoked Turkey Slices	5
167	Applegate Turkey Bacon	5

8.1.3. *Strategy 2.* Here is the suggested placement of items in refrigerators R_1, R_2, and R_5 after applying strategy 2, i.e., for configuration $a = 2$.

Table 5-1: R_1's Menu Item (under configuration $a = 2$)

Item i	Menu Item	Initial Refrigerator $R_{\#}$
50	Pennys Salsa Medium 16oz	2
51	Pennys Salsa Mild 16oz	2
52	Neato Bur Albuquerque	2
53	Neato Bur Paradise	2

54	Neato Bur Seven Layer	2
55	Neato Bur South of Border	2
56	Neato Bur Spicy Potato	2
57	Neato Bur Thai Delight	2
58	Neato Hummus Chipotle	2
65	Better Bean Hummus Cup 2oz	2
117	BelGioioso Asiago Wedge	5
118	BelGioioso Mascarpone Belgioioso	5
119	BelGioioso Parmesan Shredded Cup	5
120	BelGioioso Parmesan Wedge	5
121	BelGioioso Romano Shredded Cup	5
122	BelGioioso Romano Wedge	5
123	BelGioioso Snacking Mozzarella	5
124	Best Yet Amer Singles 12oz	5
125	Best Yet Colby Jack Shred 8oz	5
126	Field Roast Chao Cheese Original	5
159	Beechers Mac and Cheese Gluten Free	5
160	Beechers Mac and Cheese Worlds Best	5
163	Applegate Chicken Maple Sausag	5
164	Applegate Oven Roasted Chicken	5
165	Applegate Sliced Ham	5
166	Applegate Smoked Turkey Slices	5
167	Applegate Turkey Bacon	5
168	Field Roast Apple Sage Sausage	5
169	Field Roast Brkfst Mple Sausage	5
170	Field Roast Frankforter Sausage	5
171	Field Roast Fruffalo Wing	5
172	Field Roast Lentil Sage Deli	5
173	Field Roast Mex Chiptle Sausage	5
174	Field Roast Smked Tmto Deli Sliced	5

Table 5-2: R_2's Menu Item (under configuration $a = 2$)

Item i	Menu Item	Initial Refrigerator $R_{\#}$
4	Wilcox HB Egg 2Pk Cage Free	1
6	Hillshire SP Genoa Salame	1
7	Hillshire SP Hot Calabrese	1
8	Hillshire SP Ital Dry Salame	1
9	Sabra Hummus Garlic and Pretzel	1
10	Sabra Hummus Red Pepper	1
20	Teriyaki Beef Meal	1
21	Teriyaki Chicken Meal	1
48	G2G Bar Choc Banana	1
49	G2G Bar Peanut Butter & Jelly	1
59	Marukome Miso Organic 13.2oz	2
60	Marukome Miso Organic LS 13.2oz	2
61	Columbus Pastrami Presliced 6oz	2
62	Olli Pepperoni Presliced 4oz	2
63	Columbus Charcuterie Tasting Board 4oz	2
66	Divina Olives Blue Cheese Stuffed 4.6oz	2

67	Divina Olives Castelvetrano 4.6oz	2
68	Divina Olives Kalamata Pitted 4.2oz	2
69	GeradDom Wild Coho Smoked Salmon	2
70	SarbarGoMed Hummus	2
71	SarbarGoMed Hummus Rstd Garlic	2
72	SarbarGoMed Hummus Rstd Pep	2

Table 5-3: R_5's Menu Item (under configuration $a = 2$)

Item i	Menu Item	Initial Refrigerator $R_{\#}$
3	Kraft Big Cheese Bar Colby Jack	1
5	Columbus Snack Tray Italian Dry Salami	1
12	Egg Hard Boiled Individual	1
13	BBQ Turkey Meatballs	1
14	Chicken Potsticker Salad	1
15	Corn Salad	1
16	Pesto Tortellini	1
17	Quinoa Tabbolouleh	1
18	Roasted Broccoli Salad	1
19	Tropical Fruit Salad	1
22	Vegan Chana Masala	1
23	DM2Go Fresh Cut Fruit	1
24	DM2Go Chia Pudding Strawberry	1
25	DM2Go Fresh Crudite	1
26	DM2Go Fresh Grapes	1
27	DM2Go Moto Hawaiian Fried Rice	1
28	DM2Go Chicken Kale Caesar Salad	1
29	DM2Go Farmers Market Salad	1
30	DM2Go Greek Salad	1
31	DM2Go Chicken Shawarma	1
32	DM2Go Ham Havarti Croissant Sandwich	1
33	DM2Go Mediterranean Chickpea Sandwich	1
34	DM2Go Peanut Butter and Jam Sandwich	1
35	DM2Go Turkey and Cheddar Sandwich	1
127	Galbani Mozzarella Loaf 16oz	5
128	Kraft Amer Singles 8oz	5
133	Laura Chenel Goat Plain Log	5
134	Laural Chenel Garlic & Chive Goat Log 4oz	5
135	Organic Creamery Feta Crumble Cup	5
153	Boursin Cheese Garlic Herb 5oz	5
154	Daiya Vegan Cream Cheeese Strawberry 8oz	5
155	Daiya Vegan Cream Cheese Plain 8oz	5
156	Daiya Vegan Sliced Cheese	5
157	Philadelphia Cream Cheese Low Fat	5
158	Philadelphia Cream Cheese Original	5
161	Buddig Ham 2oz	5
162	Oscar M Deli Roasted Turkey 9oz	5
175	Uptons Natural Seitain Chorizo	5
176	Uptons Natural Seitan Bacon	5

8.1.4. **Economic Perspective on Item Placement.*

From an *economic* perspective, the food items in the FPF section and the frozen meals stored in R_1 are *substitutes*[6]. This means that people tend to buy one or the other, but not both. In other words, those waiting in line to buy items from the FPF section rarely have the opportunity to purchase items from the R_1 area. Therefore, upon reviewing the tables of items in each refrigerator after our strategy adjustments above, we can observe that most of the frozen meal products from R_1(like DM2GO) have been moved to R_2 and R_5.

From another economic perspective, we need to consider the concept of *limited attention*[5]. In DM's R_1, most of the stored items are frozen meals for lunch and dinner, whereas the FPF section sells ready-to-eat products that can also be served as lunch or dinner. When you are waiting in line and considering your lunch/dinner, you are unlikely to be interested in the frozen meals in R_1; you are just thinking about when it will be your turn. However, after you have purchased food from the FPF area and satisfied your immediate needs, you might start thinking about getting a heated meal for your next meal. Therefore, it might be a good idea to place these meals in R_2 and R_5.

Based on the above discussion, our strategy helps to increase the exposure of items in R_1 from an economic perspective, giving customers more opportunities to purchase food from R_1.

8.2. Specific Placement of items in R_1, R_2, R_5 under B_1(optimal configuration $a = 1$).

8.2.1. *The table below shows the 17 shelves in R_1, R_2, and R_5, as well as the lengths and widths of the 17 shelves, along with the number of distinct items we plan to place on each shelf to ensure the relative fairness of our item placement.*

Shelf (j)	Length (L_j)	Width (W_j)	# Items we plan to place
1	120	52	4
2	120	52	4
3	120	52	4
4	120	52	4
5	180	52	6
6	180	52	6
7	180	52	6
8	160	45	5
9	160	45	5
10	160	45	6
11	160	45	6
18	320	45	10
19	200	45	5
20	200	45	5
21	200	45	5
22	200	45	5
23	320	45	9

8.2.2. *Code 1: Specific Placement of items in R_1, namely shelves $j = 1, 2, \cdots, 7$.*

```
# Item data: here (x,y,z) implies (item i,item i's length,item i's width)
# For example, (3,7,20) implies item 3's length is 7cm, width is 20cm
items = [
    (3, 7, 20), (12, 5, 5), (51, 16, 16), (52, 12, 19), (53, 12, 19),
    (55, 12, 19), (56, 12, 19), (57, 16, 16), (58, 16, 16), (59, 10, 6),
    (60, 10, 6), (61, 14, 23), (63, 25, 35), (65, 10, 10), (66, 8, 8),
```

```
    (67, 8, 8), (68, 9, 9), (117, 17, 8), (122, 15, 10), (126, 10, 9),
    (133, 8, 17), (134, 7, 13), (154, 12, 12), (155, 12, 12),(156, 10, 9),
    (168, 15, 10), (169, 15, 17),(170, 15, 17),(171, 20, 2),(172, 15, 15),
    (173, 20, 14), (174, 12, 12), (175, 12, 14), (176, 14, 6)
]

# The length of Shelves in R1, we have a total of 7 shelves in R1
# with lengths 120, 120, 120, 120, 180, 180, 180(cm) separately
shelves = [120, 120, 120, 120, 180, 180, 180]

# Number of items to be placed on each shelf.
# Followed by the "fair exposure" in our paper.
# Distribute 34 items into 7 shelves:
# 4 types on the first three shelves, 6 types on other shelves
shelf_configurations = [4, 4, 4, 4, 6, 6, 6]

# Our Item placement methods in Section 4.3
def place_items_on_shelves(shelves, items, configurations):
    results = []  # List to store the placement results
    item_index = 0  # Start from the first item

    # Iterate over each shelf and its corresponding number of items to be
        placed
    for shelf_length, num_items in zip(shelves, configurations):
        # Ensure that the index does not exceed the length of the items
            list
        if item_index + num_items > len(items):
            num_items = len(items) - item_index
            # Prevent index out of range

        # Retrieve items to be placed on the current shelf
        current_items = items[item_index:item_index + num_items]
        item_index += num_items

        # Initially place each item in 1 column
        shelf_usage = [(item_id, 1) for item_id, _, _ in current_items]
        total_length = sum(item_length * 1 for _, item_length, _ in
            current_items)
        remaining_length = shelf_length - total_length

        # Use the remaining length to try to evenly increase the number of
            columns
        all_added = False
        while remaining_length > 0 and not all_added:
            all_added = True
            for i, (item_id, cols) in enumerate(shelf_usage):
                item_length = current_items[i][1]
                if item_length <= remaining_length:
                    shelf_usage[i] = (item_id, cols + 1)  # Increase by
                        one column
                    remaining_length -= item_length
                    all_added = False  # At least one item successfully
                        added

```

```
        # Append the shelf placement results to the overall results list
        results.append((shelf_usage, remaining_length, current_items))

    return results

# Width of the shelf, they were the same.
shelf_width = 52

# Calculate the number of each item that can be placed on rows
def calculate_item_rows(items, shelf_width):
    results = []
    for item_id, _, width in items:
        count = shelf_width // width
        # Use integer division to calculate the number of items that can be placed on each shelf
        results.append((item_id, count))
    return results

# Place items and calculate rows
placement_results = place_items_on_shelves(shelves, items, shelf_configurations)

# Output the results
for index, (usage, remaining, current_items) in enumerate(placement_results):
    print(f"Shelf {index + 1} (Remaining length: {remaining}):")
    for (item_id, count), (_, item_length, item_width) in zip(usage, current_items):
        # Output a_{ij}
        print(f"  a_({item_id},{index+1}) = {count}")
        # Calculate b_{ij} for each shelf
        b_ij = shelf_width // item_width
        print(f"  b_({item_id},{index+1}) = {b_ij}")
    print()
```

Output for Code 1.

```
Shelf 1 (Remaining length: 0):
  a_(3,1) = 3
  b_(3,1) = 2
  a_(12,1) = 3
  b_(12,1) = 10
  a_(51,1) = 3
  b_(51,1) = 3
  a_(52,1) = 3
  b_(52,1) = 2

Shelf 2 (Remaining length: 4):
  a_(53,2) = 3
  b_(53,2) = 2
  a_(55,2) = 2
  b_(55,2) = 2
  a_(56,2) = 2
  b_(56,2) = 2
```

```
  a_(57,2) = 2
  b_(57,2) = 3

Shelf 3 (Remaining length: 4):
  a_(58,3) = 3
  b_(58,3) = 3
  a_(59,3) = 2
  b_(59,3) = 8
  a_(60,3) = 2
  b_(60,3) = 8
  a_(61,3) = 2
  b_(61,3) = 2

Shelf 4 (Remaining length: 0):
  a_(63,4) = 2
  b_(63,4) = 1
  a_(65,4) = 3
  b_(65,4) = 5
  a_(66,4) = 3
  b_(66,4) = 6
  a_(67,4) = 2
  b_(67,4) = 6

Shelf 5 (Remaining length: 0):
  a_(68,5) = 3
  b_(68,5) = 5
  a_(117,5) = 3
  b_(117,5) = 6
  a_(122,5) = 3
  b_(122,5) = 5
  a_(126,5) = 2
  b_(126,5) = 5
  a_(133,5) = 2
  b_(133,5) = 3
  a_(134,5) = 3
  b_(134,5) = 4

Shelf 6 (Remaining length: 0):
  a_(154,6) = 3
  b_(154,6) = 4
  a_(155,6) = 2
  b_(155,6) = 4
  a_(156,6) = 3
  b_(156,6) = 5
  a_(168,6) = 2
  b_(168,6) = 5
  a_(169,6) = 2
  b_(169,6) = 3
  a_(170,6) = 2
  b_(170,6) = 3
```

```
Shelf 7 (Remaining length: 8):
  a_(171,7) = 2
  b_(171,7) = 26
  a_(172,7) = 2
  b_(172,7) = 3
  a_(173,7) = 2
  b_(173,7) = 3
  a_(174,7) = 2
  b_(174,7) = 4
  a_(175,7) = 2
  b_(175,7) = 3
  a_(176,7) = 1
  b_(176,7) = 8
```

8.2.3. *Code 2: Specific Placement of items in R_2, namely shelves $j = 8, 9, 10, 11$.*

```python
# Define the items data with length and width
items_data = [
    (23, 15, 15), (24, 9, 9), (25, 14, 14), (26, 9, 9), (27, 13, 19),
    (28, 19, 15), (29, 19, 15), (30, 19, 15), (31, 45, 45), (32, 19, 15),
    (33, 30, 45), (34, 19, 33), (35, 19, 33), (48, 8, 14), (49, 7, 13),
    (50, 12, 12), (54, 12, 19), (62, 14, 23), (69, 15, 25), (70, 15, 15),
    (71, 20, 20), (72, 15, 15)
]

# Define the length of the shelves
shelves = [160, 160, 160, 160]

# Number of item types to be placed on each shelf
C2 = [5, 5, 6, 6]

# Place items on shelves
placement_results = place_items_on_shelves(shelves, items_data, C2)

# Define the width of the shelf
shelf_width = 45

# Output the results for placement and row calculation
for index, (usage, remaining, current_items) in enumerate(
   placement_results):
    print(f"Shelf {index + 8} (Remaining length: {remaining}):")
    for (item_id, count), (_, item_length, item_width) in zip(usage,
       current_items):
        # Output a_{ij}
        print(f"  a_({item_id},{index+8}) = {count}")
        # Calculate b_{ij} for each shelf
        b_ij = shelf_width // item_width
        print(f"  b_({item_id},{index+8}) = {b_ij}")
    print()
```

Output for Code 2.

```
Shelf 8 (Remaining length: 2):
```

```
  a_(23,8) = 3
  b_(23,8) = 3
  a_(24,8) = 3
  b_(24,8) = 5
  a_(25,8) = 3
  b_(25,8) = 3
  a_(26,8) = 2
  b_(26,8) = 5
  a_(27,8) = 2
  b_(27,8) = 2

Shelf 9 (Remaining length: 1):
  a_(28,9) = 2
  b_(28,9) = 3
  a_(29,9) = 2
  b_(29,9) = 3
  a_(30,9) = 1
  b_(30,9) = 3
  a_(31,9) = 1
  b_(31,9) = 1
  a_(32,9) = 1
  b_(32,9) = 3

Shelf 10 (Remaining length: 1):
  a_(33,10) = 2
  b_(33,10) = 1
  a_(34,10) = 2
  b_(34,10) = 1
  a_(35,10) = 1
  b_(35,10) = 1
  a_(48,10) = 2
  b_(48,10) = 3
  a_(49,10) = 2
  b_(49,10) = 3
  a_(50,10) = 1
  b_(50,10) = 3

Shelf 11 (Remaining length: 1):
  a_(54,11) = 3
  b_(54,11) = 2
  a_(62,11) = 2
  b_(62,11) = 1
  a_(69,11) = 2
  b_(69,11) = 1
  a_(70,11) = 2
  b_(70,11) = 3
  a_(71,11) = 1
  b_(71,11) = 2
  a_(72,11) = 1
  b_(72,11) = 3
```

8.2.4. *Code 3: Specific Placement of items in R_5, namely shelves $j = 18, 19, \cdots, 23$.*

```
# Define the items data with length and width
items_data = [
    (4, 17, 12), (5, 16, 20), (6, 13.5, 13), (7, 13.5, 13), (8, 13.5, 13),
    (9, 10, 10), (10, 10, 10), (13, 13, 19), (14, 12, 15), (15, 12, 15),
    (16, 12, 15), (17, 17, 12), (18, 12, 15), (19, 12, 15), (20, 13, 19),
    (21, 13, 19), (22, 13, 19), (118, 12, 12),(119, 12, 12),(120, 20, 14),
    (121, 10, 10), (123, 19, 38),(124, 10, 9),(125, 14, 21),(127, 15, 20),
    (128, 10, 9), (135, 12, 12),(153, 14, 14),(157, 10, 11),(158, 12, 6),
    (159, 18, 12), (160, 10, 8),(161, 14, 18),(162, 20, 13),(163, 20, 11),
    (164, 12, 15), (165, 12, 18), (166, 9, 19), (167, 30, 13)
]

# Define the length of the shelves
shelves = [320, 200, 200, 200, 200, 320]

# Number of item types to be placed on each shelf
C3 = [10, 5, 5, 5, 5, 9]

# Define the width of the shelf
shelf_width = 45

# Place items on shelves
placement_results = place_items_on_shelves(shelves, items_data, C3)

# Output the results for placement and row calculation
for index, (usage, remaining, current_items) in enumerate(
    placement_results):
     print(f"Shelf {index + 18} (Remaining length: {remaining}):")
     for (item_id, count), (_, item_length, item_width) in zip(usage,
         current_items):
          # Output a_{ij}
          print(f"  a_({item_id},{index+18}) = {count}")
          # Calculate b_{ij} for each shelf
          b_ij = shelf_width // item_width
          print(f"  b_({item_id},{index+18}) = {b_ij}")
     print()
```

Output for Code 3.

```
Shelf 18 (Remaining length: 2.5):
  a_(4,18) = 3
  b_(4,18) = 3
  a_(5,18) = 3
  b_(5,18) = 2
  a_(6,18) = 3
  b_(6,18) = 3
  a_(7,18) = 2
  b_(7,18) = 3
  a_(8,18) = 2
  b_(8,18) = 3
  a_(9,18) = 3
```

```
  b_(9,18) = 4
  a_(10,18) = 2
  b_(10,18) = 4
  a_(13,18) = 2
  b_(13,18) = 2
  a_(14,18) = 2
  b_(14,18) = 3
  a_(15,18) = 2
  b_(15,18) = 3

Shelf 19 (Remaining length: 2):
  a_(16,19) = 3
  b_(16,19) = 3
  a_(17,19) = 3
  b_(17,19) = 3
  a_(18,19) = 3
  b_(18,19) = 3
  a_(19,19) = 3
  b_(19,19) = 3
  a_(20,19) = 3
  b_(20,19) = 2

Shelf 20 (Remaining length: 10):
  a_(21,20) = 3
  b_(21,20) = 2
  a_(22,20) = 3
  b_(22,20) = 2
  a_(118,20) = 3
  b_(118,20) = 3
  a_(119,20) = 3
  b_(119,20) = 3
  a_(120,20) = 2
  b_(120,20) = 3

Shelf 21 (Remaining length: 1):
  a_(121,21) = 4
  b_(121,21) = 4
  a_(123,21) = 3
  b_(123,21) = 1
  a_(124,21) = 3
  b_(124,21) = 5
  a_(125,21) = 3
  b_(125,21) = 2
  a_(127,21) = 2
  b_(127,21) = 2

Shelf 22 (Remaining length: 4):
  a_(128,22) = 4
  b_(128,22) = 5
  a_(135,22) = 4
```

```
  b_(135,22) = 3
  a_(153,22) = 3
  b_(153,22) = 3
  a_(157,22) = 3
  b_(157,22) = 4
  a_(158,22) = 3
  b_(158,22) = 7

Shelf 23 (Remaining length: 2):
  a_(159,23) = 3
  b_(159,23) = 3
  a_(160,23) = 3
  b_(160,23) = 5
  a_(161,23) = 2
  b_(161,23) = 2
  a_(162,23) = 2
  b_(162,23) = 3
  a_(163,23) = 2
  b_(163,23) = 4
  a_(164,23) = 2
  b_(164,23) = 3
  a_(165,23) = 2
  b_(165,23) = 2
  a_(166,23) = 2
  b_(166,23) = 2
  a_(167,23) = 2
  b_(167,23) = 3
```

8.2.5. *The following table shows the specific arrangement of items obtained through the codes above. Note that a_{ij} represents the number of columns item i can occupy on shelf j, and b_{ij} represents the number of rows item i can occupy on shelf j. Here we suppose $c_{ij} = 1$, but the manager can determine determine the number of stacks in which the i^{th} item can be placed on the j^{th} shelf vertically.*

Menu Item	i (#item)	j (#shelf)	a_{ij} (#columns)	b_{ij} (#rows)
Kraft Big Cheese Bar Colby Jack	3	1	3	2
Egg Hard Boiled Individual	12	1	3	10
Pennys Salsa Mild 16oz	51	1	3	3
Neato Bur Albuquerque	52	1	3	2
Neato Bur Paradise	53	1	3	2
Neato Bur South of Border	55	2	2	2
Neato Bur Spicy Potato	56	2	2	2
Neato Bur Thai Delight	57	2	2	3
Neato Hummus Chipotle	58	2	3	3
Marukome Miso Organic 13.2oz	59	2	2	8
Marukome Miso Organic LS 13.2oz	60	3	2	8
Columbus Pastrami Presliced 6oz	61	3	2	2
Columbus Charcuterie Tasting Board 4oz	63	3	2	1
Better Bean Hummus Cup 2oz	65	3	3	5
Divina Olives Blue Cheese Stuffed 4.6oz	66	3	3	6

Divina Olives Castelvetrano 4.6oz	67	4	2	6
Divina Olives Kalamata Pitted 4.2oz	68	4	3	5
BelGioioso Asiago Wedge	117	4	3	6
BelGioioso Romano Wedge	122	4	3	5
Field Roast Chao Cheese Original	126	4	2	5
Laura Chenel Goat Plain Log	133	5	2	3
Laural Chenel Garlic & Chive Goat Log 4oz	134	5	3	4
Daiya Vegan Cream Cheeese Strawberry 8oz	154	5	3	4
Daiya Vegan Cream Cheese Plain 8oz	155	5	2	4
Daiya Vegan Sliced Cheese	156	5	3	5
Field Roast Apple Sage Sausage	168	6	2	5
Field Roast Brkfst Mple Sausage	169	6	2	3
Field Roast Frankforter Sausage	170	6	2	3
Field Roast Fruffalo Wing	171	6	2	26
Field Roast Lentil Sage Deli	172	6	2	3
Field Roast Mex Chiptle Sausage	173	7	2	3
Field Roast Smked Tmto Deli Sliced	174	7	2	4
Uptons Natural Seitain Chorizo	175	7	2	3
Uptons Natural Seitan Bacon	176	7	1	8
DM2Go Fresh Cut Fruit	23	8	3	3
DM2Go Chia Pudding Strawberry	24	8	3	5
DM2Go Fresh Crudite	25	8	3	3
DM2Go Fresh Grapes	26	8	2	5
DM2Go Moto Hawaiian Fried Rice	27	8	2	2
DM2Go Chicken Kale Caesar Salad	28	9	2	3
DM2Go Farmers Market Salad	29	9	2	3
DM2Go Greek Salad	30	9	1	3
DM2Go Chicken Shawarma	31	9	1	1
DM2Go Ham Havarti Croissant Sandwich	32	9	1	3
DM2Go Mediterranean Chickpea Sandwich	33	10	2	1
DM2Go Peanut Butter and Jam Sandwich	34	10	2	1
DM2Go Turkey and Cheddar Sandwich	35	10	1	1
G2G Bar Choc Banana	48	10	2	3
G2G Bar Peanut Butter & Jelly	49	10	2	3
Pennys Salsa Medium 16oz	50	10	1	3
Neato Bur Seven Layer	54	11	3	2
Olli Pepperoni Presliced 4oz	62	11	2	1
GeradDom Wild Coho Smoked Salmon	69	11	2	1
SarbarGoMed Hummus	70	11	2	3
SarbarGoMed Hummus Rstd Garlic	71	11	1	2
SarbarGoMed Hummus Rstd Pep	72	11	1	3
Wilcox HB Egg 2Pk Cage Free	4	18	3	3
Columbus Snack Tray Italian Dry Salami	5	18	3	2
Hillshire SP Genoa Salame	6	18	3	3
Hillshire SP Hot Calabrese	7	18	2	3
Hillshire SP Ital Dry Salame	8	18	2	3
Sabra Hummus Garlic and Pretzel	9	18	3	4
Sabra Hummus Red Pepper	10	18	2	4
BBQ Turkey Meatballs	13	18	2	2
Chicken Potsticker Salad	14	18	2	3

Corn Salad	15	18	2	3
Pesto Tortellini	16	19	3	3
Quinoa Tabbolouleh	17	19	3	3
Roasted Broccoli Salad	18	19	3	3
Tropical Fruit Salad	19	19	3	3
Teriyaki Beef Meal	20	19	3	2
Teriyaki Chicken Meal	21	20	3	2
Vegan Chana Masala	22	20	3	2
BelGioioso Mascarpone Belgioioso	118	20	3	3
BelGioioso Parmesan Shredded Cup	119	20	3	3
BelGioioso Parmesan Wedge	120	20	2	3
BelGioioso Romano Shredded Cup	121	21	4	4
BelGioioso Snacking Mozzarella	123	21	3	1
Best Yet Amer Singles 12oz	124	21	3	5
Best Yet Colby Jack Shred 8oz	125	21	3	2
Galbani Mozzarella Loaf 16oz	127	21	2	2
Kraft Amer Singles 8oz	128	22	4	5
Organic Creamery Feta Crumble Cup	135	22	4	3
Boursin Cheese Garlic Herb 5oz	153	22	3	3
Philadelphia Cream Cheese Low Fat	157	22	3	4
Philadelphia Cream Cheese Original	158	22	3	7
Beechers Mac and Cheese Gluten Free	159	23	3	3
Beechers Mac and Cheese Worlds Best	160	23	3	5
Buddig Ham 2oz	161	23	2	2
Oscar M Deli Roasted Turkey 9oz	162	23	2	3
Applegate Chicken Maple Sausag	163	23	2	4
Applegate Oven Roasted Chicken	164	23	2	3
Applegate Sliced Ham	165	23	2	2
Applegate Smoked Turkey Slices	166	23	2	2
Applegate Turkey Bacon	167	23	2	3