Optimization for Recipe-based, Diet-planning Inventory Management

by

Sheng Kang

B.E. Information Security

Nanjing University of Posts and Telecommunications (2009)

SUBMITTED TO THE SCHOOL OF ENGINEERING

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE IN COMPUTATION FOR DESIGN AND OPTIMIZATION

AT THE

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

SEPTEMBER 2010

Author ______________________________

Sheng Kang
School of Engineering
August 2, 2010

Certified by ______________________________

Richard Larson
Mitsui Professor of Engineering Systems
Thesis Supervisor

Accepted by ______________________________

Karen Willcox
Associate Professor of Aeronautics and Astronautics
Codirector, Computation for Design and Optimization Program

Optimization for Recipe-based, Diet-planning Inventory Management

by

Sheng Kang

Submitted to the School of Engineering

in Partial Fulfillment of the Requirements for the Degree of

Master of Science in Computation for Design and Optimization

On August 2nd, 2010

Abstract

This thesis presents a new modeling framework and research methodology for the study of recipe-based, diet-planning inventory management. The thesis begins with an exploration on the classic optimization problem – the diet problem based upon mixed-integer linear programming. Then, considering the fact that real diet-planning is sophisticated as it would be planning recipes rather than possible raw materials for the meals. Hence, the thesis develops the modeling framework under the assumption that given the recipes and the different purchasing options for raw materials listed in the recipes, examine the nutrition facts and calculate the purchasing decisions and the yearly optimal minimum cost for food consumption. This thesis further discusses the scenarios for different groups of raw materials in terms of shelf-timing difference. To model this inventory management, the modeling implementation includes preprocess part and the optimization part: the formal part involves with conversion of customized selection to quantitative relation with stored recipes and measurement on nutrition factors; the latter part solves the cost optimization problem.

Thesis Supervisor:

Richard Larson

Mitsui Professor of Engineering Systems

Director, Center for Engineering Systems Fundamentals

Acknowledgements

I would like to express my deep and sincere gratitude to my supervisor, Professor Richard Larson, Ph.D, Mitsui Professor of Engineering Systems, Director of the Center for Engineering Systems Fundamentals. He has been supportive since the days I started my study at MIT. Richard has supported my not only by providing insight thought through his expertise, but also academically and emotionally encouraged me with his strong support and tremendous enthusiasm. His constructive comments and feedbacks on my research and writing helped me significantly during the through the tough road to finish this thesis. Thanks to him I had the opportunity to design and build a practical model that could be used in real life! He helped me come up with the thesis topic and guided me over almost a year of development. This is the happiest time I have spent when study abroad and thanks for giving me moral support and the freedom I needed to move on especially in the difficult times during thesis work.

My next gratitude goes to Professor Karen Willcox. I remember her kind help and advice during the meeting regarding my research and thesis work in the fall semester, 2009. The discussion that day gave me a more comprehensive understanding about the schedule, process and useful resources to smooth the research work. I also want to thank Laura Koller for her sincere help in the CDO program.

Many people who work and study in the CDO and ORC have taught and helped me immensely at MIT. Thanks for all the friends here who have enriched my life and made it more enjoyable than I could ever expected. Let me thank my friends who have been studied abroad together with me: Jingting Zhou, Shuo Tian, Ying Xiong, Jingxi Wang, Naijun Wen, Zhibo Sun, Daizhuo Chen, Bin Huang, Liwei He, Sandeep Tiwari, Meng Li, Thai Dung, Venugopalan Raghavan, Vinh Nguyen. You let me feel that we are a great family and I have enjoyed the time we spent together in Boston or during the trip! Thank Li Liang and Cheng Ju for being helpful seniors. Thank Min Ding, Yan Chen, Yajing Tu, Ling Li, Xiang Zhou, Sisi Ni, Chong Hou and Miao Yu for the first and warm Spring Festival that I had overseas, we had great Chinese food and watched the Gala Evening that day!

Final acknowledgement should go out to my family for their unconditional love, trust and encouragement. They have supported me whenever I needed it and cherished with me every great moment. Thank you so much for always standing by my side and believing in me. Without you I would not be able to come all my way today.

Table of Contents

Figures

Tables

Chapter 1. Introduction

The traditional and classic diet problem is to find the cheapest combination of foods that will satisfy all the daily nutritional requirements of people, a problem, which can be formulated as a linear program. The approach is improved later by solving it using mixed integer programming (MIP) because many products are sold only in units. Unfortunately, the practice of the results indicates that the cost is usually higher than the expectation. One possible explanation is that the problem is formulated under the circumstance that foods can be simply combined to serve the people, which is not true. Hence, in this paper, we formulate the problem in a recipe-based model.

Chapter 2 will give a brief description of the problem statement, the goals in each phase of the project and the implementation platform as well as the assumptions of the project implementation.

Chapter 3 will focus on preprocess in the model. In this paper, the phases to solve the problem will be separated to two parts. The first is the preprocess part that works on the input and provides intermediate variables that would be passed to the second phase.

Chapter 4 would state the optimization method used in this paper, which can generate the results of purchasing decisions and compute the yearly minimum cost for the selected menu.

Chapter 5 would introduce how we implement the approaches explained in Chapter 3 and Chapter 4 by Excel vba and Add-ins component.

Chapter 6 would give a set of results and their corresponding observations based on the interpretation of single test result and comparison of the results.

Chapter 7 will provide with a summary conclusion and discuss the possible future improvement.

Chapter 2. Project Description

The classical diet problem is to obtain the minimum cost of the daily diet given the nutrition constraints, which is a linear problem. The assumption for the diet problem is simple, the users can purchase any quantity of the raw materials for diet planning, which is usually not realistic. As the measurement of each item might be quite different: some can be purchased by a specific amount required, like bok choy can be bought by *lbs* and some can only be purchased on integral multiples of fixed number. For example, milk is stored in some specific containers, like 1 quart bottle, ½ gallon bottle, 8 ounce box and so on. You cannot checkout half of a ½ gal bottle of milk in a market. This is the reason that when using the classic diet problem model to minimize the cost, the real cost is always higher than the expectation. Therefore, we need to first transform the originally simple linear problem into an MIP problem.

Figure 2.1 Measurement comparison of bok choy and milk

Second, since the combination given by the result of the diet problem might not be ideal for cooking, implying that the combination itself might be meaningless and unrealistic. The fact that people tend to prepare meals by recipes, rather than simply put them together, means we need to consider the practical matter that the diet-planning shall be based on recipes. The recipes would build a connection between food to be consumed and the quantities needed in the selected menu. This idea was presented in the British Journal of Nutrition (1995) by Pingsun Leung, Kulavit Wanitprapha and Lynne A. Quinn.

Hence, in this paper, I would like to propose a model that first give the recipes and data for users to select, and then examine the nutritional facts in the selected menu – this would be the preprocess part. Next, to make the condition closer to the reality, I will divide the raw materials into three groups based on the shelf-timing differences. Considering the characteristics of the foods, here I use MIP to solve the problem weekly within one year eventually.

To make it clear, the implementation is composed of two phases: the first is the preprocess phase mentioned above; the second is the utilization of optimization method. Generally speaking, the purpose for the first phase is to ensure that the whole problem can satisfy the nutrition requirements; the goal for the latter phase is to generate the purchasing decision among multiple alternatives.

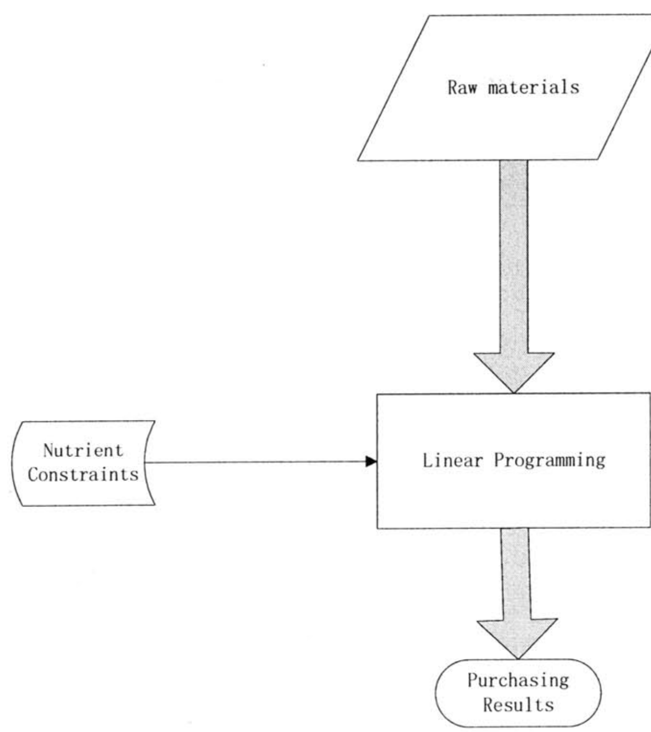

Figure 2.2 the flow chart for classic Diet Problem

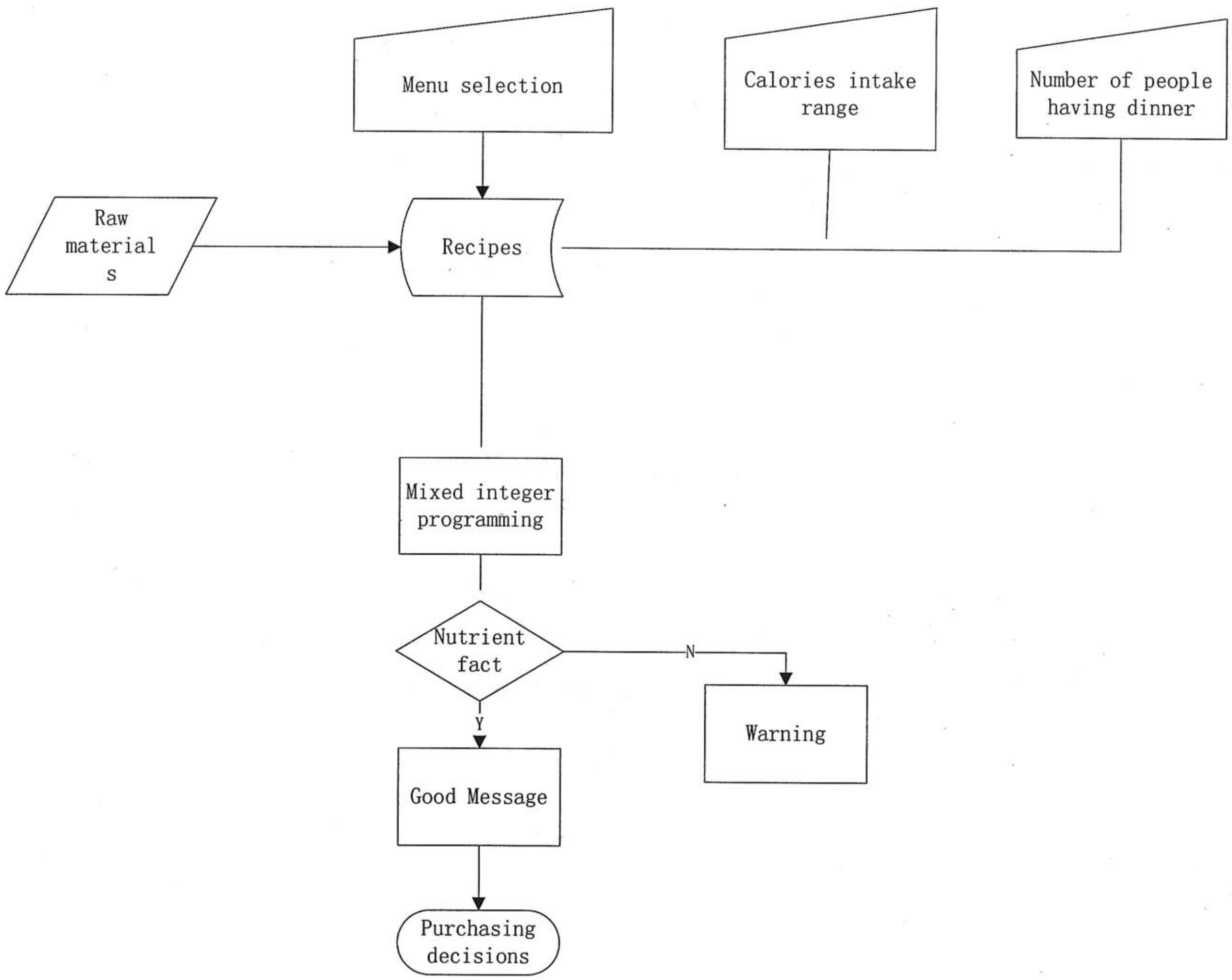

Figure 2.3 the flow chart for the recipe-based, diet-planning problem in this paper

Figure 2.2 and Figure 2.3 give a brief depiction about the differences among the two problems. Basically we connect the input and the process part through the data information in the recipe data base. Here we neglect the details for phase two, which focus on computing the optimal purchasing decisions for different groups of products based upon the shelf-timing differences.

Now it comes to the platform for implementation. I originally use *Matlab* to construct models for the two phases separately and verified the feasibility of our problems. Since we would like to provide the users with a more friendly platform to try the model themselves, I use Excel vba with an add-ins component (‘*What'sBest!*’) to give the users a direct sense of how the model works.

Chapter 3. Approach for the Diet-planning Preprocess

Although the model discussed in this paper is derived from the diet problem, the different perspective that it is looking into makes it a distinguished problem. Here we would first introduce the formulation of the classic diet problem. And then we would further develop the formula used in the preprocess part in this paper.

- **Literature Reivew**

The diet problem:

Denote x_j as the amount we purchase for food j and p_j as the corresponding unit price for food j.

Also denote c_i as the nutrient level required for the nutrient type i and a_{ij} as the amount of nutrient i per unit in food j.

As the objective is to minimize the total cost, which could be expressed in term of $\sum_{j=1}^{n} p_j x_j$. Hence, the objective becomes: $Min\ \sum_{j=1}^{n} p_j x_j = p * x$

Then consider the constraints in the linear problem, in the diet problem, it is to satisfy the minimum nutrient demand. The constraint is shown as the following formula.

$$a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_{in} = \sum_{j=1}^{n} a_{ij}x_j \geq c_i$$

If there are m nutritional requirements, then the constraints formula must hold for $i=1$ to $i=m$.

As the food amount purchased must be positive, we need to have $x_j \geq 0$.

Now, to sum up, the formulation of the diet problem shall be:

Obj: $Min \sum_{j=1}^{n} p_j x_j = p * x$

Where $\sum_{j=1}^{n} a_{ij}x_j \geq c_i$

$x_j \geq 0$

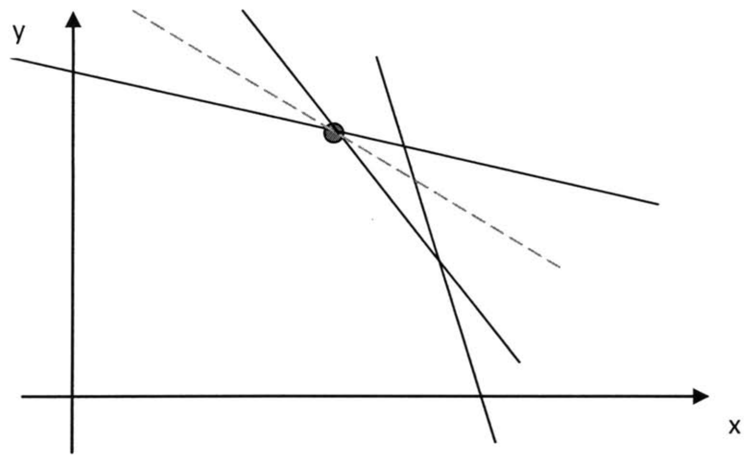

Figure 3.1 An example of two-dimensional linear programming problem

Figure 3.1 can give a brief description of the linear programming problem, this is a two-dimensional case, and multi-dimensional linear programming problem would be much more complex, but the principal would be the same. The blue dashed line indicates the slope for the two variables x and y given the objective formula. This is a feasible question that can reach its maximization objective in the red circle under five constraints.

The preprocess in this model:

While in the preprocess part mentioned in this paper, the amount of products are linked through the recipe data base with the input. Without loss of generality, let x_j still be the amount of *j*th food, and *n* would be the total number of food types. For the *i*th recipe, the amount of raw materials required in the recipe is constant given certain serving size. Let y_{ij} be the amount of *j*th food needed in the *i*th recipe and *r* be the total number of recipes in the data base.

Assume the serving size for all the recipes would be the same, then we have

$$\sum_{i=1}^{r} y_{ij} = y_j \ .$$

However, since the default serving size in the recipe data base is typically constant, say 4 as example. Then the actual consumption of the raw materials shall be the constant multiple of sum in the default recipe data base. Denote the constant as *D*.

$$\text{Then } x_j = D * y_j$$

Now the objective for the overall diet planning problem would still be to $Min \sum_{j=1}^{n} p_j x_j = p * x$.

This transformation makes the nutrient constraints shall be the same. However, since we keep the recipes as data base, the constraints could also be stored in the data base for every recipe rather than be made as a separated data base for raw materials.

Then, we need to determine the exact value of *D*. One of the very important elements in the preprocessing is to compute *D*, and this *D* would be used in the second phase of computation.

It is easy just to input the number of serving size, and work out the relation between the input and the serving number in default setting. However, the reality is, the default serving size probably is not fit for everyone. Therefore, if the serving size is too small, then we probably would not feel full after having the meals; if the serving size is too big, we might waste the food. For accuracy purpose, we need to customize the serving size, or to find the relation between the default serving size and our demand. We select one of the nutritional factor as an input and use the connection between the input and default nutrient data in the recipe to compute the *D*.

Calories, now becomes the most concerned factor in intake nutrients for most people. The measurement of calories intake is easier to understand compared to other trace nutrient elements. Make calories intake ranges as input choice, then we could obtain a range of *D*. Without loss of generality, in the project implementation, we use an approximate integer as an output of the real serving size with respect of the *D*. We later find that the difference of the calories option would make the serving size quite different.

The menu provided to the users is directly linked to the data base of the recipes. After completing the calories range intake and menu selection, we will find the nutrients intake shall be determined. How about the nutrient requirements? Although generally most dishes are designed to balance overall nutrients, yet we cannot be confirmed that the selection would not go extremes. Therefore, in the preprocess part, we would assess the other nutrition intake and examine if they satisfy the constraints obtained. In the next step, we would provide the information summary of the users' preference, whether or not one nutrient intake is low, or if all the constraints are strictly obeyed. It is the users' decision to keep some nutrient low or high, for example, low carbohydrate intake or low fat intake is reasonable.

Chapter 4. Optimization Method Based on the Recipes

After the preprocessing, every raw material is correlated with the amount worked out by the first phase, which means that the weekly food consumption quantity is known. It is obvious that for each raw material, we are faced with diversified options in the market -- price, volume, quality, brands -- they all affect our purchasing decision. Since we need to minimize the cost for meal preparation, the price and volume become more important than quality or brands (unless the requirements to the raw materials are specified). For every single purchase, we would like to meet the demand level and choose the one with lowest cost. This idea makes a term essential here -- *unit price*. Quoted from wiki, "the unit price is a valuation method for buyers who purchase in bulk", that means the cost per unit or per measure. Holding the volume or measure constant, to save money, we would like to select the lower unit price. People may be familiar with the term "family size" if they pay attention to the products with large volume in the market. It is a common sense that a big family tends to choose items with "family size" on the package when shopping in the market. The reason lie under this behavior is that the unit price is generally lower than purchasing small packages for the family members. Nevertheless, the unit price is only part of the factor, cooperated with its corresponding measure and the demand of the item acquired by phase one, the purchasing decision for every product given a known demand can then be obtained.

Another condition in this modeling process is that we make the classification of raw materials needed by their shelf-timing difference. Some products are easy to expire and perish, like vegetables, meat and milk. We group those foods as short shelf-timing products. While some products can be kept for around one year or even longer, like rice, salt, flour and so on. We group those as long shelf-timing products. Here we use one year as the shelf life for all long shelf-timing product. In addition, to make the condition closer to the reality, we group items like garlic, yogurt, pie shells as the ones with shelf period between the former two groups, namely moderate shelf-timing products. In this model, those moderate shelf-timing products would expire in 30 days after first use.

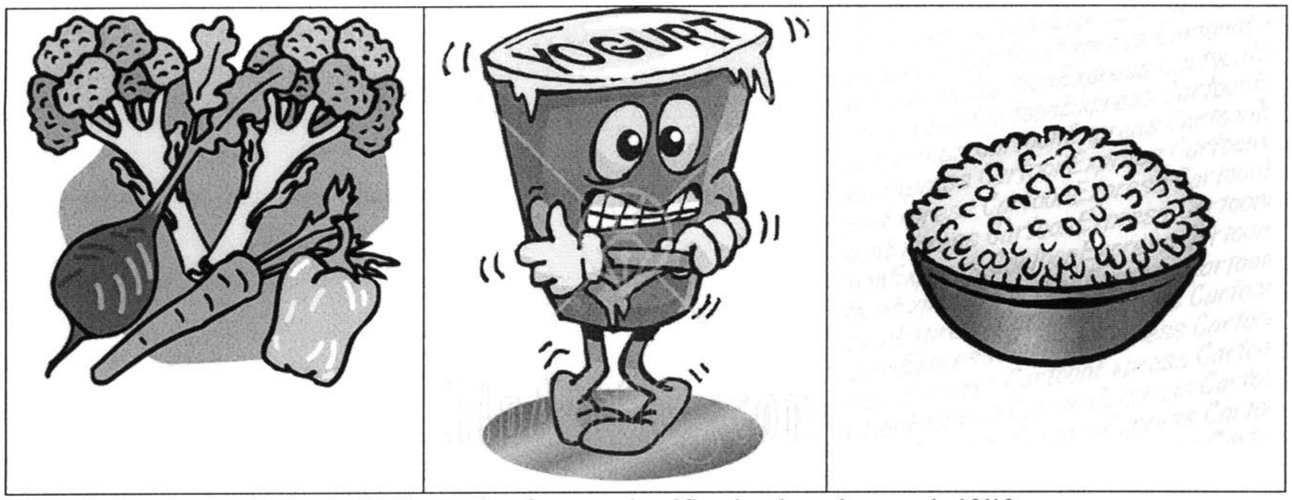

Figure 4.1 Example of group classification based upon shelf life

(from left to right: short group, moderate group, long group)

For short shelf-timing products, these goods shall be purchased every week. The purchasing decision will be the same for each week. For long shelf-timing products, these goods can be purchased once and could last for one year. We therefore suggest the users purchase enough quantity for the entire year in the first week, though you could choose to buy part of them later. For moderate shelf-timing products, conditions become relatively complex. After purchasing, the moderate items can cover at least four-week time, and then expire. It would save more if we compute the overall demand for four weeks and then use this amount to replace weekly demands. With the change of demands, the decision for moderate shelf-timing products would be based on a repeating period with four weeks. Another idea is to cover the next two days so that the materials could be made full use of. However, the reality is that the materials of the additional two days cannot be determined without knowing the users' decision. As we consider the selected menu as an integrated element, here we ignore the additional two days. And the moderate products can only be passed to the next three weeks after the first use.

Upon the analysis of the different groups, we now continue to introduce the formula used to compute the purchasing decision.

What known in the second phase are demands of each item per week, price and measure options for every raw materials.

Notations:

Let d_j be the demand for the jth item, p_{ij} be the price of the ith option for the jth product, v_{ij} be the measure of the ith option for the jth product, x_{ij} be the number we need to buy on the ith option for the jth product,

Therefore, the formula would be

$$\text{Obj: } Min \sum_i p_{ij}x_{ij}, \forall j,$$

$$\text{Where } \sum_i v_{ij}x_{ij} \geq d_j$$

$$x_{ij} \geq 0$$

$$x_{ij} \in integers$$

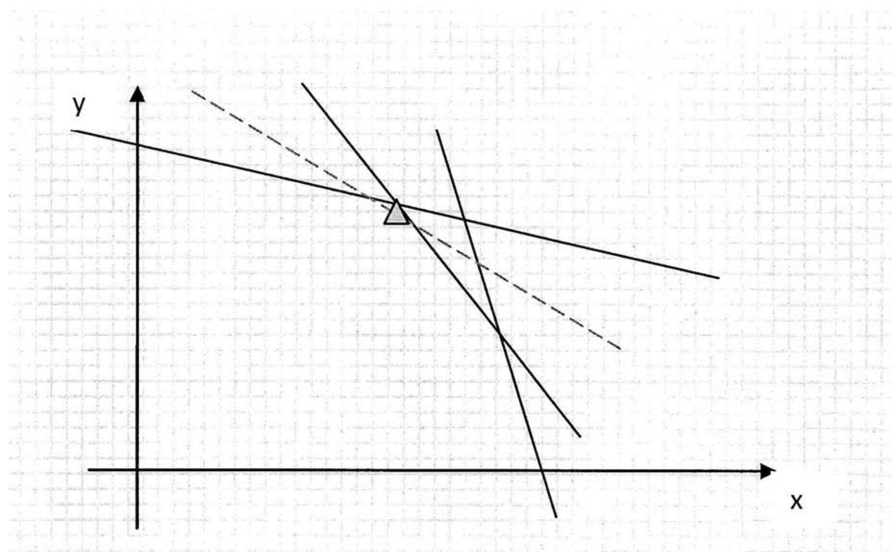

Figure 4.2 An example of two-dimensional mixed integer programming problem

Compared to the Figure 3.1 that depicted about linear programming in Chapter 3, we could find that there are several difference in mixed integer programming for the same problem.

a. The results are no longer infinite in the above example. There are a number of integer points indicating integer results in the solution set. In the two-dimensional example, the feasible areas are the points on the grey grid.
b. For the same problem, the minimization objective that can be obtained would be larger than that of the linear programming problem while the maximization objective that can be obtained would be smaller than that of the linear programming problem.

Figure 3.1 and Figure 4.2 are the same maximization problem, however, the optimal result can only be obtained in the yellow triangle because all results for x and y have to be integers. The difference would be more obvious if the grid pace for integers is larger.

If for every j, the solution to the above formula can be obtained, then the overall minimum cost can be acquired. Because x_{ij} is integer, thus this problem becomes a MIP question. Notice that the p_{ij} would be a limited set for every jth item, this data would be stored in a spreadsheet that contains the price information for every product.

Chapter 5. Implementation

The platform for implementation here is Microsoft Excel. The reason for selecting Excel as the platform is that currently Excel is the most popular and flexible business modeling software and project implentation environment. I used both functions inside Excel and *vba* to program the two phases. Moreover, a free trial version of Add-in component "*What'sBest!*" for linear programming needs to be installed.

A brief introduction of the function of *What'sBest!* is as follows: it is an add-in to Excel that could enable the construction of large scale optimization models. The free trial of the add-in can be downloaded through http://www.lindo.com/index.php?option=com_content&view=article&id=36&Itemid=21. The free trial software will not expire. However, the capacity for the constraints, adjustable and integers setting in the linear programming model are limited. In the project I worked, the variables and constraints are workable under the limit, but the setting for number of integers cannot be satisfied. I provided the solution to force the variables to be integers by adding a *vba* code.

In the meanwhile of keeping the original characteristics of the project mentioned above, to simplify the implementation and make it easier for users to manipulate, here a menu is provided for the users to choose their weekly selection from a given recipe data base. Everyday the users could choose two main courses, one soup and one dessert for dinner. The options are not necessarily to be all filled, but warning will be generated once a cell is left blank to remind the users about blank content in the menu selection.

First step in the implementation is to consider the placement of the data. One Excel document is consisted of different spreadsheets. In the implementation development, I created four sheets, which are *menu_sel*, *menu*, *recipe* and *price*. The *menu_sel* provides the user with an interface to give selections as the input, including the weekly menu, number of people having the dinner and also the calories intake range.

The *menu* contains the three lists of dish names, which are named as *ranges*. On the right side of the lists, there are the cells indicating the count of total selection in a week time. These numeric data would be linked to sheet 3, which is sheet *recipe*.

The *recipe* sheet contains the names of the dishes and their corresponding recipes. For each dish, it also links the count of total selections in a week. In the bottom of this sheet, it will compute the overall nutrient intake given the menu selection. The output -- real serving size -- generated by phase one is also listed here.

The sheet *price* is the sheet that shows the output upon the completion of the phase two. In this sheet, I use three colors to indicate different groups of raw materials: red represents short shelf-timing group; green represents long shelf-timing group; orange represents moderate shelf-timing group. For every food required, on the right side, first is the measurement for the product, then three row of cells containing the price and measure information per unit item for the product, the third row is for output decision of the product. To its right side, there is the demand constraint. The known variable of weekly demand is linked to the former sheets. In this sheet, the cost on every single purchase for each item will be worked out in the cells listed under "cost".

A screenshot of sheet *menu_sel* is shown below:

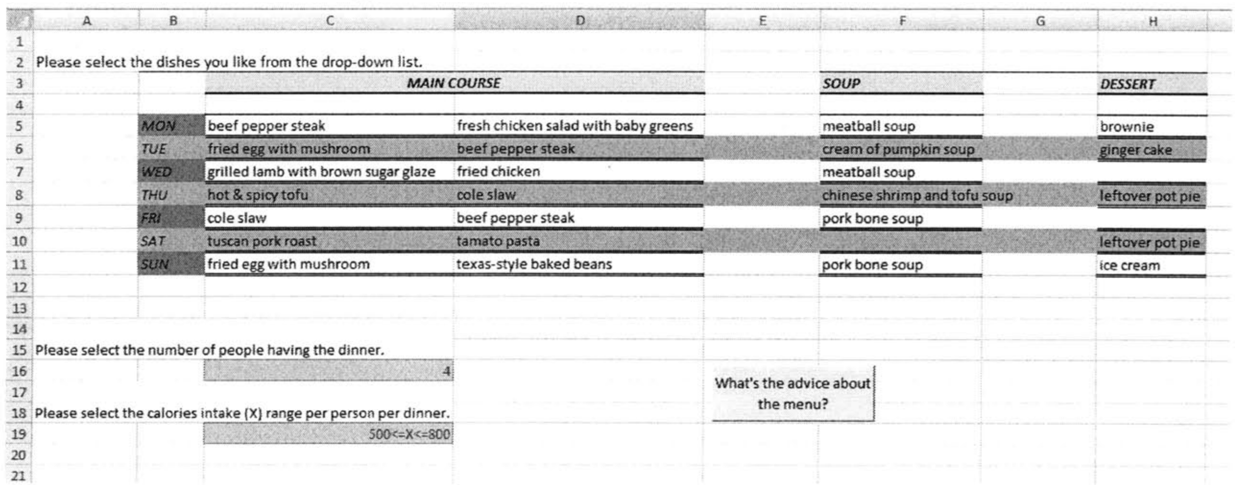

Figure 5.1 Screentshot of sheet *menu_sel*

In this screenshot you could find that most of the options are already filled. The menu selection is shown in the blue and white area. For every option, you could find a drop-down list: these lists are connected with a range. In Excel, the range is a group of cells in a worksheet that have been selected or highlighted. Here I named three ranges which are stored in the sheet *menu*. They are the name lists of main course, soup and dessert. It will first examine if there are cells left blanks after selection, then it would evaluate the nutrient fact of the selected menu, either generate warning messages or pop a congratulation message box. Moreover, it would compute the real serving size given the default recipes data base (default serving number is 4 for every recipe). After a click on the bottom, the algorithm will be triggered.

A screenshot of sheet *menu* is as follows:

	A	B	C	D	E	F
1		*SUM PER WEEK*		*SUM PER WEEK*		*SUM PER WEEK*
2	***MAIN COURSE***		***SOUP***		***DESSERT***	
3						
4	baked fish	0	chinese shrimp and tofu soup	1	brownie	1
5	beef pepper steak	3	cream of pumpkin soup	1	cinnamon rolls	0
6	cole slaw	2	meatball soup	2	ginger cake	1
7	fast & friendly meatballs	0	mushroom soup	0	ice cream	1
8	fresh chicken salad with baby greens	1	pork bone soup	2	leftover pot pie	2
9	fried chicken	1	tomato egg soup	0	potato chips	0
10	fried egg with mushroom	2			pudding	0
11	grilled lamb with brown sugar glaze	1				
12	hot & spicy tofu	1				
13	noodles	0				
14	pork and bamboo shoots	0				
15	quick chicken stew	0				
16	rice	0				
17	shrimp-stuffed eggplant rolls	0				
18	tomato pasta	0				
19	texas-style baked beans	1				
20	tuscan pork roast	1				
21						

Figure 5.2. Screenshot of sheet *menu*

A screenshot for part of the sheet *recipe* is as follows:

		package	lb	lb	unit	lb	package	package	unit	
	SUM PER WEEK	bacon	bamboo shoots	beef fillet	bell pepper	boneless pork	bread	cake	carrots	ch
AIN COURSE										
aked fish	0									
eef pepper steak	3			1.00						
ole slaw	2				1.00				2.00	
ast & friendly meatballs	0									
esh chicken salad with baby greens	1				1.00					
ied chicken	1									
ied egg with mushroom	2									
rilled lamb with brown sugar glaze	1									
ot & spicy tofu	1				1.00					
oodles	0					0.20				
ork and bamboo shoots	0		0.50			0.50				
uick chicken stew	0				1.00				1.00	
ce	0									
hrimp-stuffed eggplant rolls	0									
omato pasta	0	0.50								
exas-style baked beans	1				2.00					
scan pork roast	1				1.00	0.50				
OUP										
hinese shrimp and tofu soup	1									
ream of pumpkin soup	1									

Figure 5.3 Screenshot of sheet *recipe*

A screenshot for part of the sheet *price* is as follows:

A	B	C	D	E	F	G	H	I	J	K
Price										***cost***
bacon	package		5.99	1.99	3					0
			1	1	1					
			0.0	0.0	0.0		0	=>=	0.00	
bamboo shoots	lb		1.26							0
			1							
			0.0				0	=>=	0.00	
beef fillet	lb		9.99	10.49	10.79					49.95
			0.5	0.5	0.5					
			5.0	0.0	0.0		2.5	>=	2.25	
bell pepper	unit		4.99	0.99						5.94
			3	1						
			0.0	6.0	0.0		6	>=	5.25	
boneless pork	lb		9.98	2.99	5.24					5.98
			2	1	0.75					
			0.0	2.0	0.0		2	>=	1.13	
bread	package		2.99	3.99	1.99					0
			1	1	1					
			0.0	0.0	0.0		0	=>=	0.00	
cake	package		6.99	5.29	4.29					8.58
			1	1	1					
			0.0	0.0	2.0		2	>=	1.50	
carrots	unit		1.29	1.69	2.19					1.29
			10	12	12					

Figure 5.4 Screenshot of sheet *price*

In this screenshot, the cells with content in blue indicate the variables. These variables are the workout for a single purchase. We could see from the table that this group is the short shelf-timing product. Take bacon as an example, the explicit explanation for the cells is

5.99	1.99	3						0
1	1	1						
0.0	0.0	0.0		0	=>=		0.00	

Table 5.1. Table content for material bacon

The three top left cells are used to store the price information per unit item, the cells under the three cells are the corresponding measurements per purchase. We find that they are all ones, meaning that bacon shall be purchased by integer multiples of one single package. The 0 in yellow is actually the sum product of the three cells in the second row and the third row. The 0.00 in green is the demand linked to the former sheet *recipe* with respect to the group classification. The purchasing decisions for different group are different: the short shelf-timing group is correlated to the weekly demand, while the moderate shelf-timing group is correlated to the demand that covers four weeks.

Chapter 6. Results and Observations

In this chapter, we will mainly discuss different results and observations through running this program. First, we will introduce how to manipulate it.

In the following screenshot, it shows the sheet *menu_sel* with all the input contents blank. Generally, we need to fill most contents from the drop-down list by clicking the right side of the input cells.

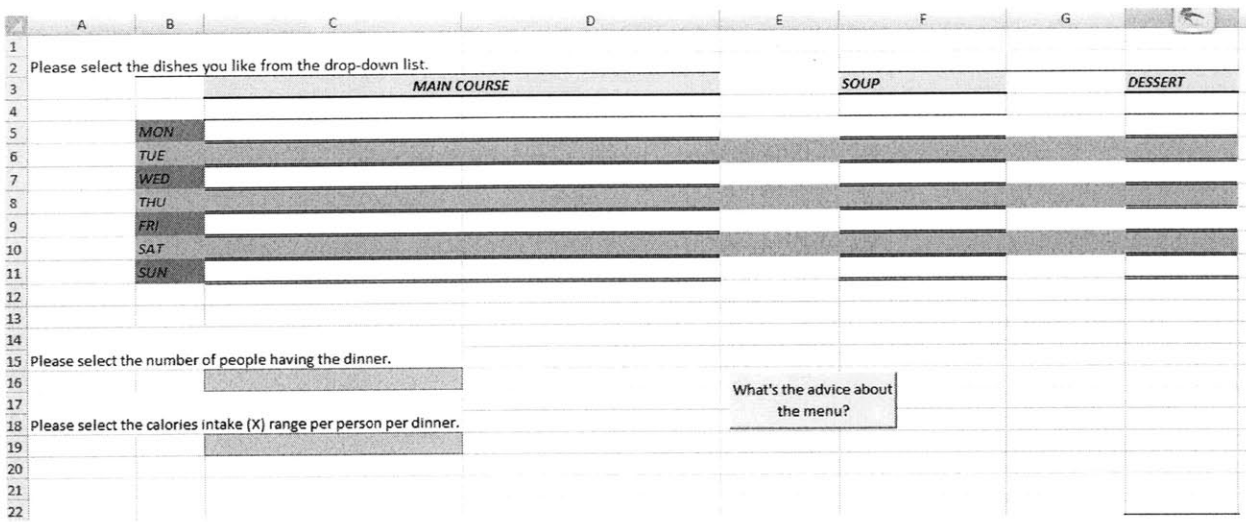

Figure 6.1 Screenshot of spreadsheet *menu_sel* with input areas all cleared

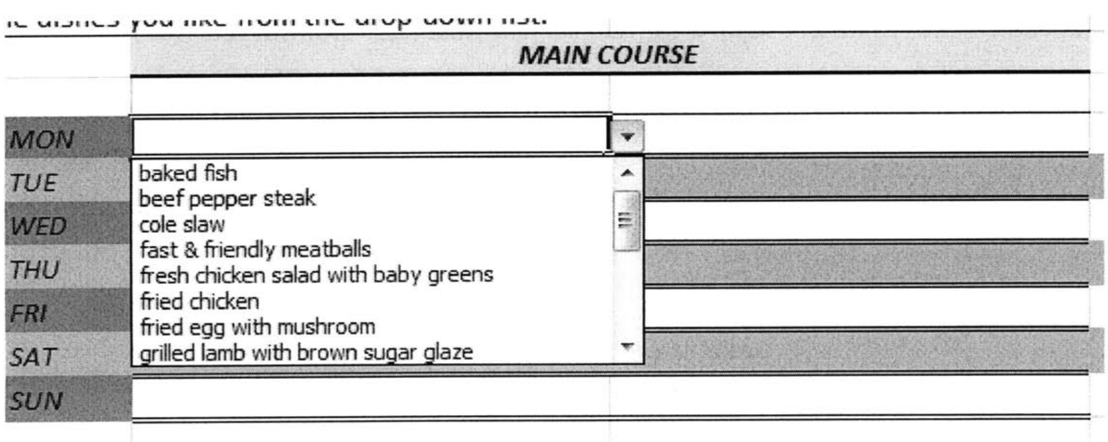

Figure 6.2 Screenshot of the drop-down list for the menu selection

In the tool menu bar, select the Add-Ins, you could see a bottom on the right side next to the "=" with a red target design. This is the bottom for computing the second phase. The shape is . Before selection, be sure to check the toolbar menu. Under the "Calculation Options" choice, please select "Automatic". This is because a lot of the contents in the cells are calculated by Excel functions, they would not update the results when changed if it is under "Manual" option.

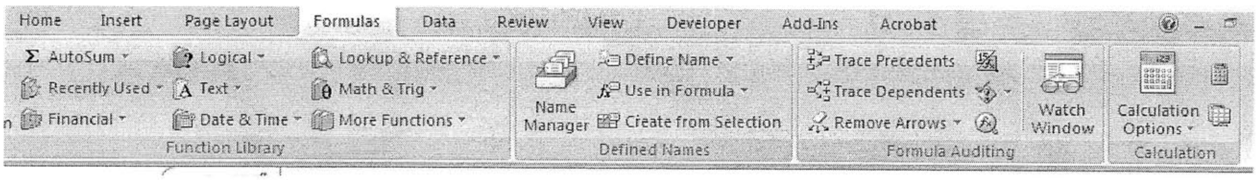

Figure 6.3 The toolbar in the Excel

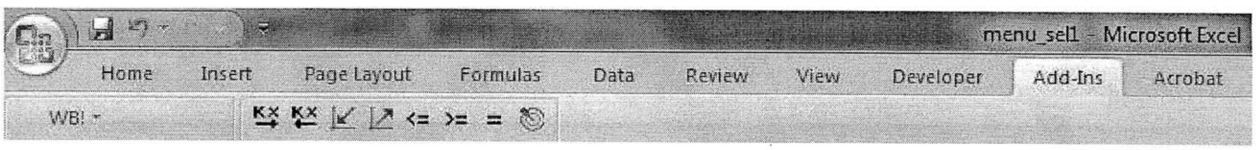

Figure 6.4 The Add-in *"What'sBest!"* with operation bottoms

Except for the menu selection, we also need to input the number of people having the dinner and the ideal selection for average calories intake per dinner. If the cell for inputting the number of people is empty, the result would be 0 for real serving size and 0 for the yearly cost.

However, if the selection for average calories intake is empty, then the default setting would be that the calories shall be higher than 500 and less than 800 per dinner. This is manipulated set in the vba code, we could also use *vba* to define other value as the default setting.

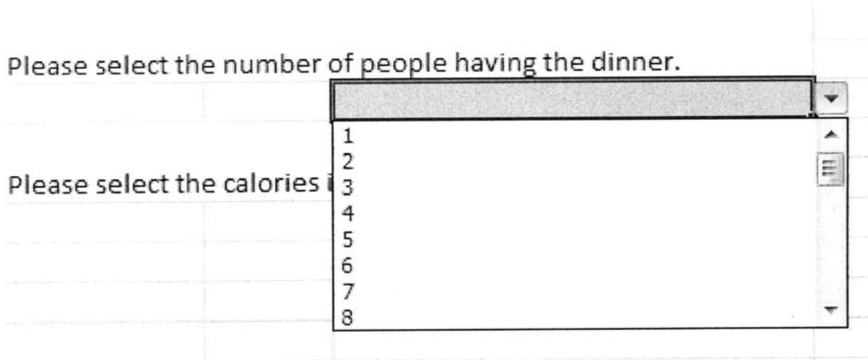

Figure 6.5 Input selection: number of people having the dinner

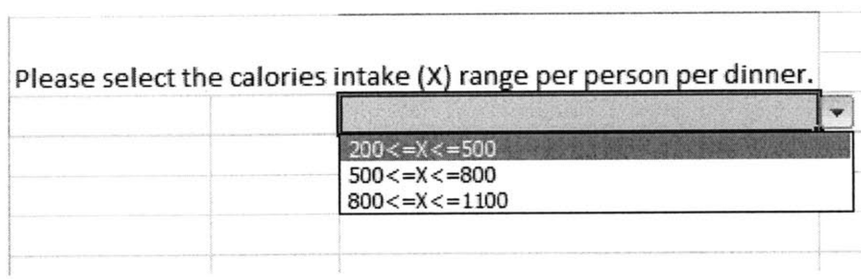

Figure 6.6 Input selection: calories intake range

To test the result of keeping some menu selections blank, we leave cell F8 and H9 blank on purpose. And other cells are all filled regularly. Then hit the execution bottom "What's the advice about the menu?" We could see a message box with content "You haven't selected the dishes in the red areas" and would give the column and row name of the cell.

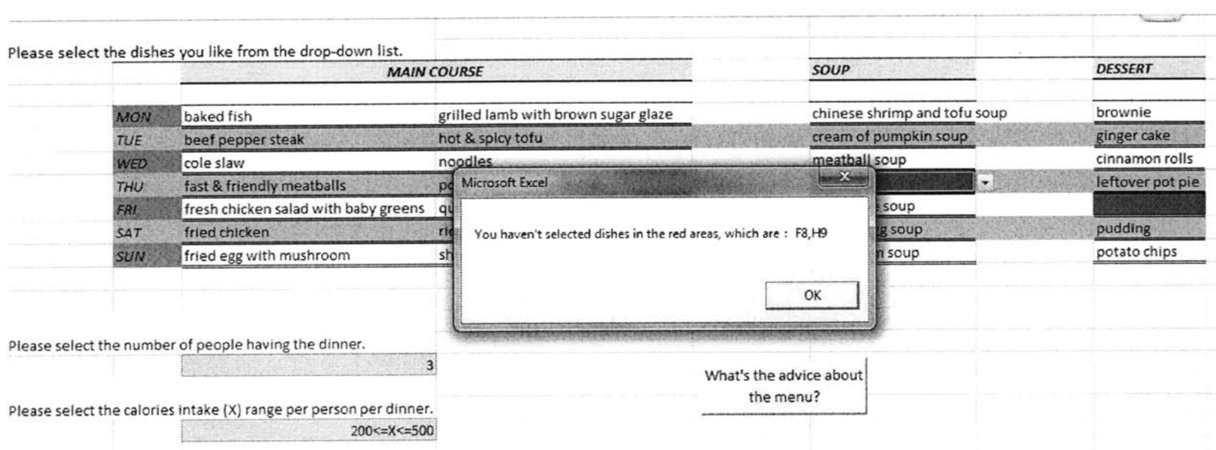

Figure 6.7 Example of warning message box after leaving areas blank

The message box will appear again with message regarding the real serving size. Also, since it will check the nutrient information about the selected dishes. It would calculate whether or not they satisfy the nutrition requirements given known calories intake range. If not, a message box with a warning, such as "The carb intake for the menu selection is relatively low" will be generated. Nowadays, many low carbohydrate intake recipe are used on purpose, therefore we do not force the customers' selection to strictly satisfy all the nutrient constraints. In this case, in order to increase the carbohydrate intake, the users could select some dishes with high proportion of carbohydrate intake, such as pasta, rice or cinnamon rolls.

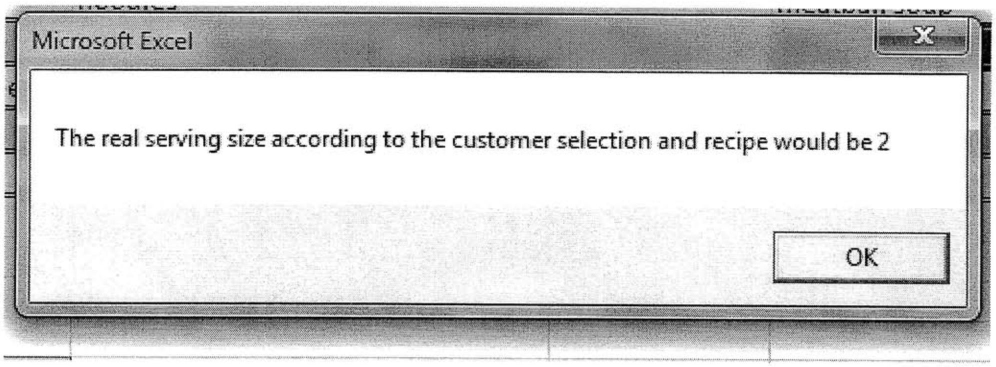

Figure 6.8 Message box with real serving size

From what Figure 6.8 and Figure 6.9 show, we could find that, probably it is because that the calories intake decision is lower than the calories contained in the recipe for each individual, the real serving size is only 2 when there are 3 people consuming the dishes.

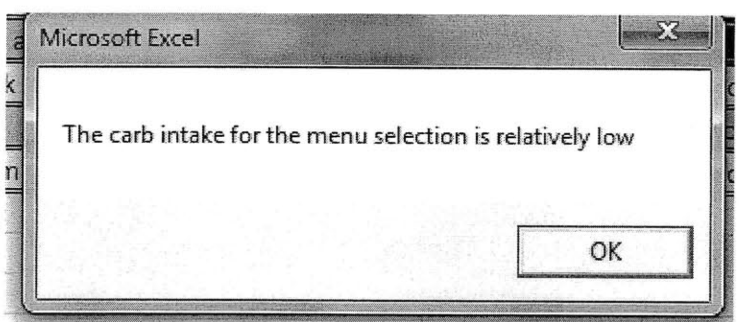

Figure 6.9 Message box with nutrient fact

While Figure 6.10 informs us that the nutrient does not completely match what be suggested by *USDA Food Guide.*

After completing the information input, when all the message boxes are shown, we could hit the bottom , which is introduced before. The results can later be found in two places: first is in the cell of K195 on the sheet "*price*", shown as Figure 6.11; the other one is located on the automatic generated sheet of summary result by the "*What'sBest!*", which is listed under "OBJECTIVE VALUE".

Yearly Cost	4888

Figure 6.10 The optimal yearly cost given selection

The following figure is a set of purchasing decisions for a certain object. Olive oil is grouped as one of the long shelf-timing products. It could be purchased once a year. Such product often has many choice options in the market with different volumes. Given the selection shown in Figure. 6.8, the demand is 52.14 oz a year. After running the algorithm to minimize the cost for purchasing olive oil, it turns out that we shall purchase the product with 101 oz as volume and \$25.99 as price. We also found that we would have 101-52.14 = 48.86 oz left. The volume we purchased is almost twice the level we need to use. However, we cannot purchase the one with volume 25.5. Though two bottles only cost \$18.98, the amount demand cannot be satisfied.

olive oil	oz		25.99	9.49	7.99					25.99
			101	25.5	17					
			1.0	0.0	0.0		101	>=	52.14	

Figure 6.11 Price and quantity areas for given product

Then we would experiment a group of tests to assess the trend of how real serving size and cost goes with the increase of the people having the dinner holding other input the same; we also would compare the results caused by different calories intake ranges.

An combined menu selection:

	MAIN COURSE		SOUP	DESSERT
MON	beef pepper steak	fresh chicken salad with baby greens	tomato egg soup	brownie
TUE	fried egg with mushroom	tomato pasta	cream of pumpkin soup	ginger cake
WED	grilled lamb with brown sugar glaze	fried chicken	meatball soup	ice cream
THU	hot & spicy tofu	cole slaw	chinese shrimp and tofu soup	potato chips
FRI	cole slaw	beef pepper steak	pork bone soup	leftover pot pie
SAT	tuscan pork roast	fast & friendly meatballs	mushroom soup	pudding
SUN	fried egg with mushroom	texas-style baked beans	cream of pumpkin soup	cinnamon rolls

Figure 6.12 An example with full selection

The above figure shows our dish selections. In terms of statistics, it is listed as follows.

	SUM PER WEEK
MAIN COURSE	
baked fish	0
beef pepper steak	2
cole slaw	2
fast & friendly meatballs	1
fresh chicken salad with baby greens	1
fried chicken	1
fried egg with mushroom	2
grilled lamb with brown sugar glaze	1
hot & spicy tofu	1
noodles	0
pork and bamboo shoots	0
quick chicken stew	0
rice	0
shrimp-stuffed eggplant rolls	0
tomato pasta	1
texas-style baked beans	1
tuscan pork roast	1
SOUP	
chinese shrimp and tofu soup	1
cream of pumpkin soup	2
meatball soup	1
mushroom soup	1
pork bone soup	1
tomato egg soup	1
DESSERT	
brownie	1

cinnamon rolls	1
ginger cake	1
ice cream	1
leftover pot pie	1
potato chips	1
pudding	1

Table 6.1 Statistics of dish selections

Then we run the program 90 times: people having dinner varied from 1 to 30, and calories intake ranges are: [200, 500], [500, 800] and [800, 1100]. Then we use *matlab* to plot the comparison results when we obtain the test results. Indicate the trend of [200, 500] calories intake by '*', the trend of [500, 800] by '◊', the trend of [800, 1100] by 'o'.

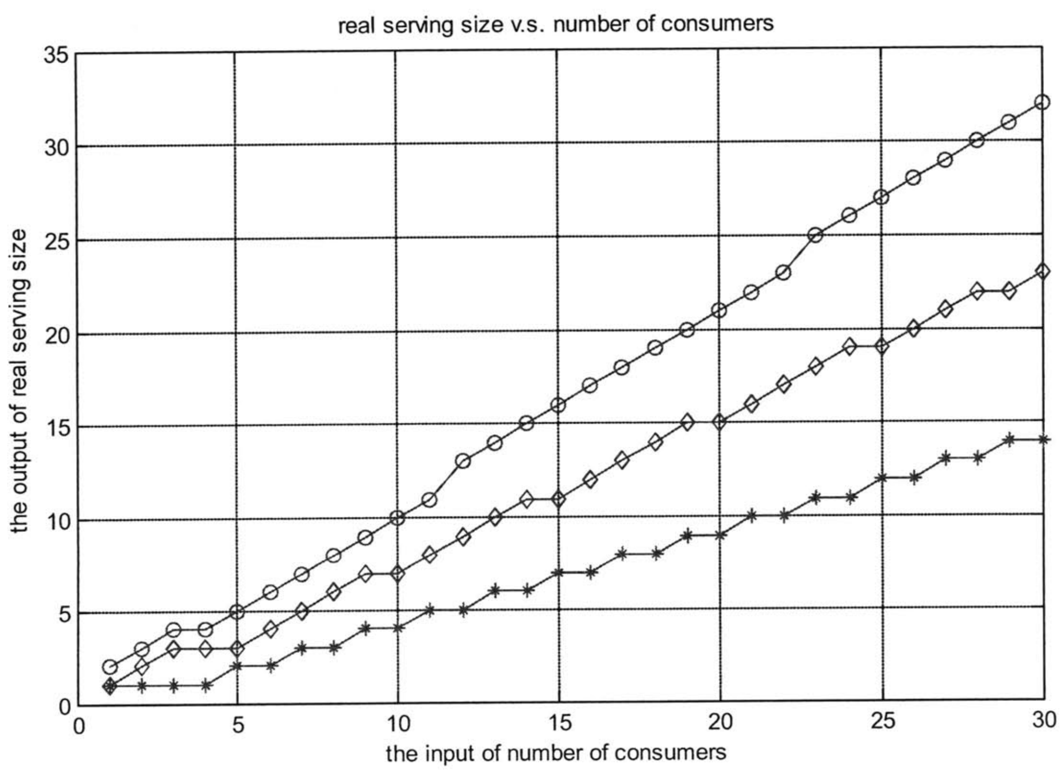

Figure 6.13 Results of real serving size

The above figure shows the trends and comparison with respect to different calories intake ranges. Several interesting observations can be seen through the figure.

1) Generally, the real serving size is going up as the number of people having dinner increases. The real serving size might stay the same if the amount prepared by integers multiples of the recipes could still satisfy the demand.
2) If we call it the plateau phase for the period that different number of people sharing the same number of real serving sizes. Then we could find that the plateau phases from the plot figure that the low calories intake is more frequently shown than that of high calories intake. One possible explanation is that if the calories intake is lower than that in the default recipe for one person, we might encounter plateau phase. However, if the calorie intake is higher than that in the default recipe, we probably would not see such phenomenon. Instead, we might could find a leap with bigger inclination. Such leap in the above figure could be found in the 'o' line, where the number of real serving size is increasing from 11 to 13 while most of the increase pace is only 1.

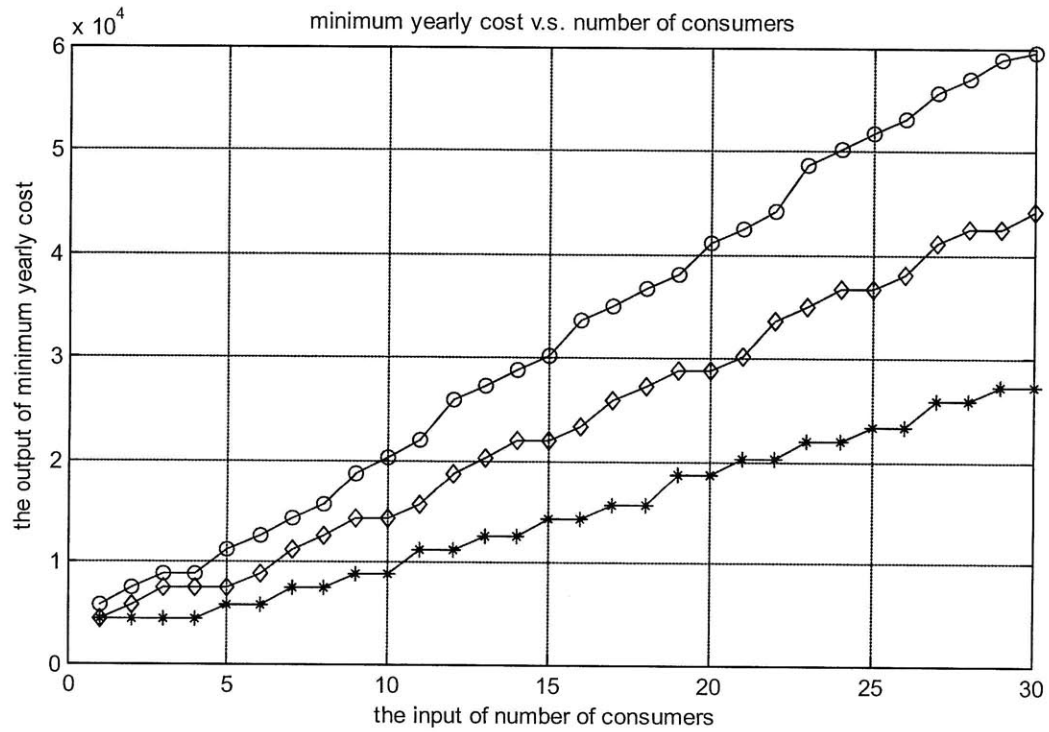

Figure 6.14 Results of optimal yearly cost

Then we observe another set of data – the minimum yearly cost for the dish selection, which is also one of the objective in the second phase of implementation. We could find that the trends are similar with the figure plotting real serving size. We could also conclude some information from this figure.

1) The yearly cost is increasing as the number of people having the dinner goes up holding the other setting the same, which is consistent with the common knowledge.
2) The slope of the trend for segments within the same line would be within a range. The general trend for high calories intake is steeper than that of a lower calories intake. To verify this observation, we would plot the trend and list the information in the following figure.

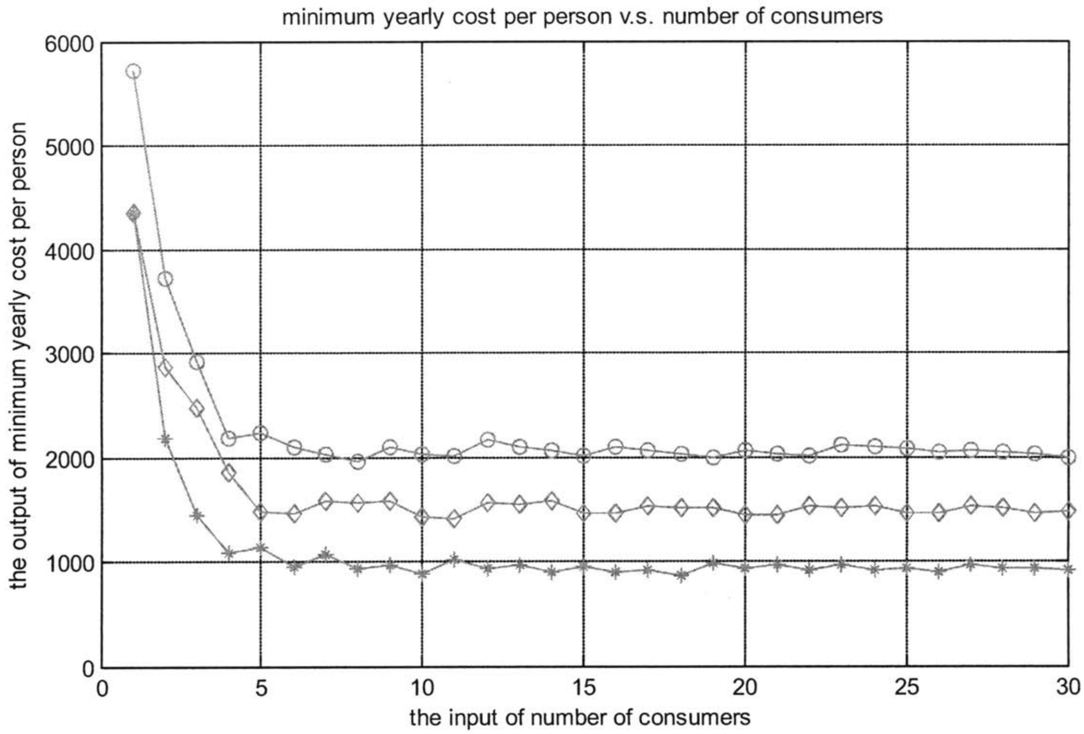

Figure 6.15 Results of optimal yearly cost per person

This figure contains the plotting for the minimum individual yearly cost given the experiment results. Similarly, we will make some safe conclusions given the observations above.

1) There are some ups and downs in the lines. The shape is not so regular. It is likely that when the serving size is increasing, at the turning point, the minimum yearly cost per person would goes up a little instead going down. Because the marginal cost for the sum might be larger than the yearly cost per person before the joint.
2) Another fun observation is that: it is highly likely that the individual cost when there is only one person having the food is highest, meaning it would be quite expensive when we live alone. If we choose to live together or to have a family, it would cost more in all but would save money when it comes to the perspective from individual spending.
3) For this dish selection, the input of 4 as people having the dishes is very critical. The trend is increasing dramatically within the range 1 to 4. The individual's cost after 4 is almost as low as it for big family like 13 or 20. This means that the family consisting

with 4 or 5 members probably would be a wise and money-saving combination. As we can also find that in the figure 21, the yearly cost for 5 people is higher than that of 4. So it might be safe to say that family with 4 members might be the best option. In the meanwhile of keeping the overall consumption low, we can also make the cost per person low.

Next, we would give the minimum value among the three setting calories intake and find the optimal value for the number of people considering to minimize the individual's cost as the objective. The data is listed in the following table.

calories intake range	people having the dinner	real serving size	total yearly cost	individual's yearly cost
[200, 500]	18	8	15594.16	866.34
[500, 800]	11	8	15594.16	1417.65
[800, 1100]	8	8	15594.16	1949.27

Table 6.2 Minimum individual's yearly cost table

1) Since the calories intake per person is increasing, it is apparently that the minimum individual's yearly cost shall be increasing. However, this conclusion does not mean that any data in low-calories intake group would be lower than that of high-calories intake. At least the maximum individual's yearly are all at first point and the value is the same.
2) Another interesting fact from this table is that though the inputs are quite different, they are all corresponding to the same real serving size, which is 8. Since the minimum cost per person would be the optimal when the waste of the food consumption only takes a small proportion. It is like the optimal result among the optimal combinations for purchasing decisions.

A repetitive menu selection:

If we have the same dishes every night. For example, here we input the baked fish as the main courses, Chinese shrimp and tofu soup as the soup selection and brownie as the dessert. Again we run the same number of trials for testing.

Plots with similar trends can be generated with the results worked out by the model. However, by comparison with the three plotting above, we could make some observations in a more general way.

1) The calories in the default setting for this menu selection is higher than that of the example for the combined selection. Because given certain calories demand, the real serving size is smaller.
2) Though the real serving size is smaller, the expense for this simple selection is higher than the above example. One possible explanation is that the cost for seafood is typically higher than that of normal meat, like chicken or pork.

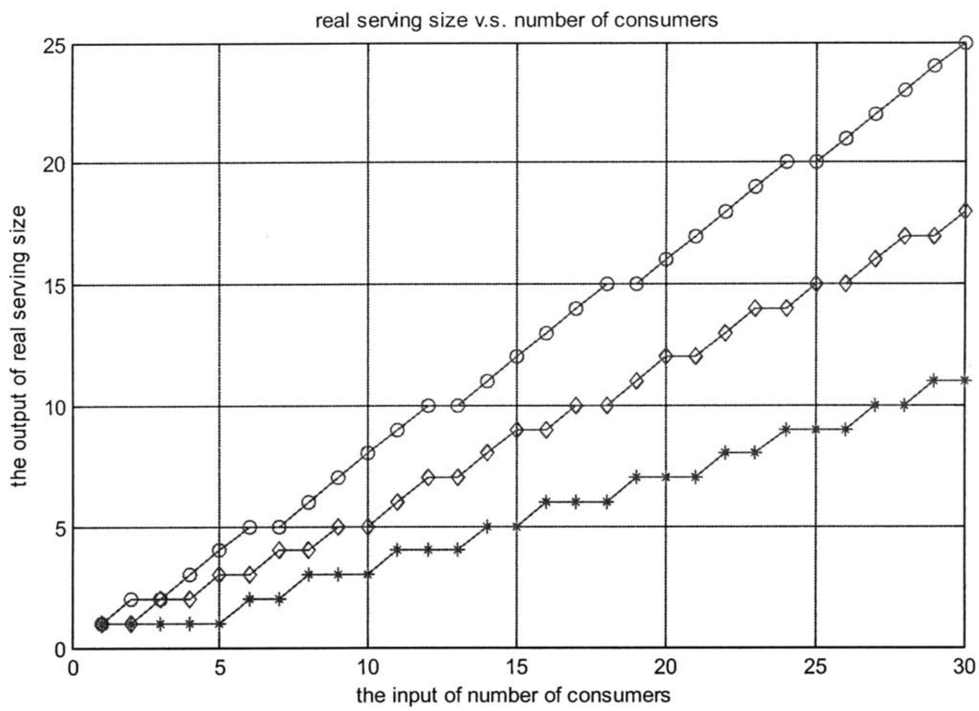

Figure 6.16 Results of real serving size (same dishes everyday)

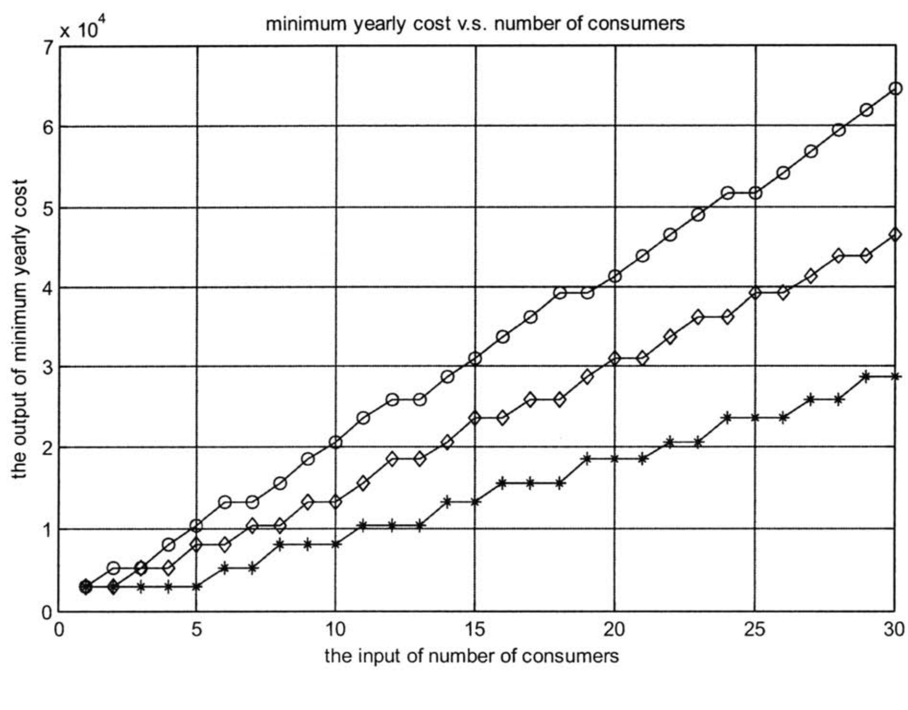

Figure 6.17 Results of minimum yearly cost (same dishes everyday)

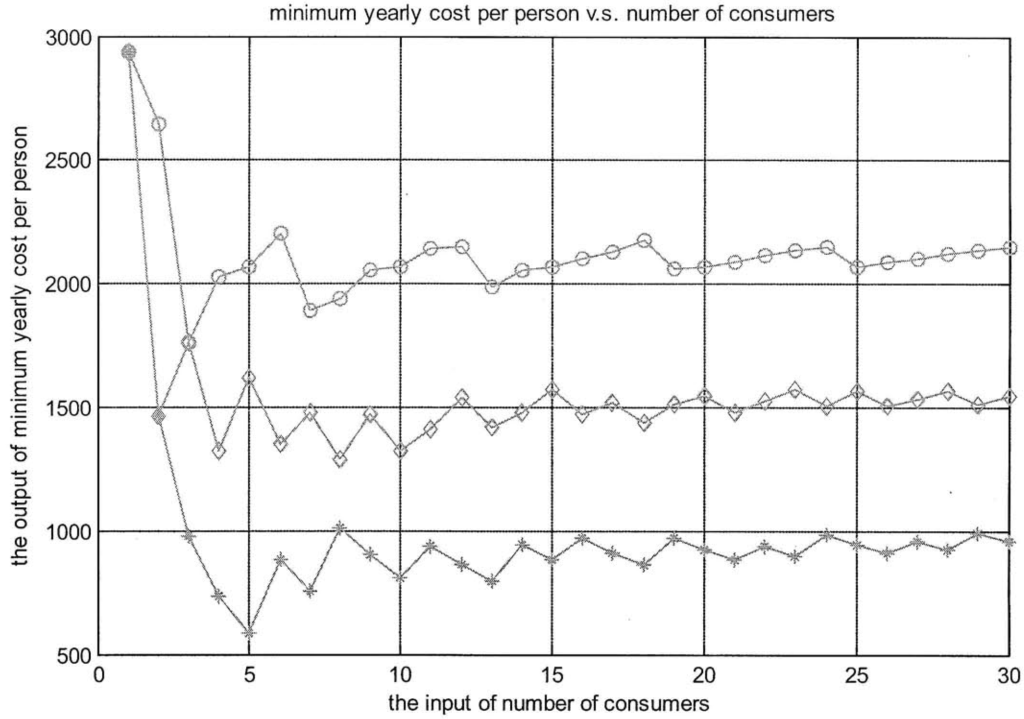

Figure 6.18 Results of minimum yearly cost per person (same dishes everyday)

We also want to know whether or not the minimum cost per person would be different compared to the above combined menu selection. A quite different result is found through the following table. The real serving size is no longer the same for the three calories intake ranges. Another interesting fact is that they reached the optimal individual's yearly cost when the number of people is not so large. Moreover, it seems there is no trend that the optimal number of people having the dinner will increase or decrease when the calories intake increases. It is likely that given an extreme condition that the consumers would have the same dishes every single day, the usage of the products might be quite different when there are more people join the dinner. It is also possible that if a family consumes simple dishes every day, taken from the perspective of individual's yearly cost, the family can reach a minimum objective with a smaller size of family members than that of the combined menu selection.

calories intake range	people having the dinner	real serving size	total yearly cost	individual's yearly cost
[200, 500]	5	1	2936.93	587.39
[500, 800]	8	4	10333.75	1291.72
[800, 1100]	3	2	5284.1	1761.37

Table 6.3 Minimum individual's yearly cost table (same dishes everyday)

Chapter 7. Conclusion

This paper provides a method and an insight view to solve such type of problems. How to deconstruct, formulate and model the problem, also it leaves space for readers to explore in case you might be interested. The program can be modified and it is easy to change the data base for the recipes and plan the meals yourself.

Planning for our daily meal consumptions not only saves us money and also make us understand more about the process of linear integer optimization problems and the importance of grouping materials. It reminds us of considering the objective of minimizing the cost from both ways: in terms of overall cost and minimum cost per person. We shall acquire some knowledge like "unit price" and "nutrient constraints" after this paper. We shall also be aware that there lies some reasons for our practical actions, such as the big family tends to purchase family-size products that are easily to used up. While for a person lives alone, he or she might only need to buy small quantity of food although the unit price per measure is always higher than that of family-size product. However, the reason for this selection is that his/her consumption demand might be low. S/he might waste most of the food when s/he only consider unit price as criterion of making decision without noticing personal need.

If we compare the results in this paper and that of the classic diet problem, we could find that is much higher when we use recipes as the data base than just use raw materials as input data base. Life is much more complex. To combine all the simple raw materials and serve people typically does not happen in real life except for some extreme conditions: like to help as many people as possible in slum areas to help them meet the minimum nutrient requirements; some on war conditions when the budget and materials are limited.

From all the tests and observations given the results, we could find that even for the optimal value, there is still some waste. As a matter of fact, since here we only design the dinner as the scenario for diet-planning, while in real life, we could plan for three meals a day. And if we could adjust the recipe or the menus, the results might be much different. Since we might take usage of the "waste" and find some feasible cooking way to make them eatable. This would certainly help eliminate the waste. We might also keep some leftover dishes to next meals and add remaining materials to serve the family.

Even in the recipe data base, it is possible to have the same calories intake but with lower overall cost for the group of people and also for each individual. As we could select the dishes either with low expense or take full usage of the materials we might need, which is, if we decide to use a raw material, we do not want to use it only once and waste most of it.

Excel code:

```
Sub run_click()
'
' run_click Macro
''''''''''''''''''''''''''''''''''''''''''''''''''''''
' check blank selection
''''''''''''''''''''''''''''''''''''''''''''''''''''''
' check if the number of blank cells is 0
Sheet1.Range("week_selection").Select
If (Selection.Cells.Count - WorksheetFunction.CountA(Selection)) > 0 Then

'if there are blank cells

Dim rng As Range
Set rng = Sheet1.Range("week_selection").SpecialCells(xlCellTypeBlanks)
rng.Select
rng.Interior.ColorIndex = 3
MsgBox "You haven't selected dishes in the red areas, which are£º " & rng.Address(0, 0)
Set rng = Nothing
End If


''''''''''''''''''''''''''''''''''''''''''''''''''''''
' calculate calores intake for regular 4 serving size selection

''''''''''''''''''''''''''''''''''''''''''''''''''''''
' calculate the real serving size

''''''''''''''''''''''''''''''''''''''''''''''''''''''
Dim cal_lower, cal_upper As Integer
Dim serving_selection As Integer
Dim calories_selection As String
Dim calories_default As Double
Dim ser_no_lower, ser_no_upper As Double

serving_selection = Sheet1.Range("c16")
calories_selection = Sheet1.Range("c19")
calories_default = Sheet3.Range("b43") / 7
```

```
If calories_selection = "" Then
MsgBox "Wrong! You haven't selected calories intake. Default Setting would be
'500<=X<=800!'"
Sheet1.Range("c19") = "500<=X<=800"

ElseIf calories_selection = "200<=X<=500" Then
cal_lower = 200
cal_upper = 500
ElseIf calories_selection = "500<=X<=800" Then
cal_lower = 500
cal_upper = 800
Else
cal_lower = 800
cal_upper = 1100
End If


ser_no_lower = cal_lower / calories_default * serving_selection
ser_no_upper = cal_upper / calories_default * serving_selection

If ser_no_upper - ser_no_lower < 1 Then
Sheet3.Range("b44") = Application.WorksheetFunction.RoundUp(ser_no_lower, 0)
Else
Sheet3.Range("b44") = Application.WorksheetFunction.RoundDown(ser_no_upper, 0)
End If


MsgBox "The real serving size according to the customer selection and recipe would be "
& Sheet3.Range("b44")

If Sheet3.Range("bn43") / Sheet3.Range("bm43") < 0.033 Then
MsgBox "The fat intake for the menu selection is relatively low"

ElseIf Sheet3.Range("bp43") / Sheet3.Range("bm43") < 0.143 Then
MsgBox "The carb intake for the menu selection is relatively low"

ElseIf Sheet3.Range("bq43") / Sheet3.Range("bm43") < 0.032 Then
MsgBox "The protein intake for the menu selection is relatively low"

Else
MsgBox "Your nutrition intake is OK! ^-^"
End If


End Sub
```

Matlab code for plotting results:

```
clear;
clc;
close all;
result = load('result_com_cost.txt');
x = result(1:end,1);

y1 = result(1:end,2);
y2 = result(1:end,6);
y3 = result(1:end,10);

yy1 = result(1:end,3);
yy2 = result(1:end,7);
yy3 = result(1:end,11);

yyy1 = result(1:end,4);
yyy2 = result(1:end,8);
yyy3 = result(1:end,12);

plot (x, y1, 'r-*', x, y2, 'r-d',x, y3, 'r-o');
grid;
xlabel('the input of number of consumers');
ylabel('the output of real serving size');
title('real serving size v.s. number of consumers');
figure;
plot(x, yy1, 'b-*', x, yy2, 'b-d',x, yy3, 'b-o');
grid;
xlabel('the input of number of consumers');
ylabel('the output of minimum yearly cost');
title('minimum yearly cost v.s. number of consumers');
figure;
plot(x, yyy1, 'g-*', x, yyy2, 'g-d',x, yyy3, 'g-o');
grid;
xlabel('the input of number of consumers');
ylabel('the output of minimum yearly cost per person');
title('minimum yearly cost per person v.s. number of consumers');
```

References

1. P. Leung, K. Wanitprapha and L. A. Quinn (1995). A recipe-based, diet-planning modelling system. *British Journal of Nutrition, 74,* 151-162. doi:10.1079/BJN19950119
2. R. Levi, M. Pal, R. Roundy and D. B. Shmoys (2005). *Integer Programming and Combinatorial Optimization.* Springer Berlin / Heidelberg. doi:10.1007/11496915_23
3. D. Bertsimas and J. N. Tsitsiklis. Introduction to Linear Optimization, Athena Scientific, 1997.
4. P. H. Zipkin. Foundations of inventory management. The McGraw-Hill Companies, Inc, 2000.
5. D. Simchi-Levi, X. Chen and J. Bramel. *Logic Of Logistics* (2nd ed.). Springer-Verlag, 2004.
6. J-K. Bok, H. Lee, J. W. Chang and S. Park (2002). Development of refinery scheduling system using mixed integer programming and expert system . *Korean Journal of Chemical Engineering, 19,* 545-551. doi:10.1007/BF02699293
7. N. Agrawal and S. A. Smith (2008). *Retail Supply Chain Management: Quantitative Models and Empirical Studies* . Springer.
8. D. Simchi-Levi, J. Bramel and X. Chen (2005). *The Logic of Logistics : Theory, Algorithms, and Applications for Logistics* (2nd ed.)
9. J. E. Russo (1977). The Value of Unit Price Information . *Journal of Marketing Research, 14*(2), 193-201.

10. B. Mittal (2006). Measuring Purchase-decision involvement. *Psychology and Marketing, 6*(2), 147-162. doi:10.1002/mar.4220060206

11. R. J. Ribando (2010). An Excel/Visual Basic for Applications (VBA) Programming Primer. Retrieved from http://faculty.virginia.edu/ribando/modules/xls/VBAPrimer.pdf