Small Group Optimization: Creating Balanced Environments for Villanova Buddies

Joseph Zinno

Villanova Buddies is a student organization at Villanova University with a mission of inclusion of individuals with developmental or intellectual disabilities. A focus of Villanova Buddies is creating Small Groups comprised of students and individuals with disabilities. Groups convene throughout the semester to create a space for friendship and social interaction. To create well-rounded spaces, the formation process of Small Groups aims to evenly distribute participants by gender and class year throughout all Small Group time slots. This is important to create diverse groups, as participating Buddies come from all different backgrounds. It is crucial that all feel comfortable, no matter their age, gender identity, or interests. This paper explores the formulation of a mathematical model using linear programming and how the results of this formulation translate into Small Groups. Multiple models are used to analyze the tradeoffs between satisfying student preferences and balancing the groups.

Introduction

Villanova Buddies is an organization that aims to promote inclusion and friendship by providing a space for students and individuals with intellectual or developmental disabilities to build relationships and socialize. Small Groups is a branch of the Villanova Buddies organization that partners six to 12 Villanova students with a group of three to five individuals with disabilities. Small Groups are a part of the Villanova Buddies organization to create smaller spaces with few people to further promote the friendships and connections initiated during Villanova Buddies' other events. Some of these events include watch parties for Villanova games, picnics on campus, or holiday parties like Friendsgiving or Valentine's Day. Small Groups meet during designated times throughout the semester. For example, the "Monday Small Group" meets one Monday evening per month throughout the semester. The time slot for each Small Group remains constant throughout the entire semester so everyone can plan accordingly to be consistently present. Having all members at all of their respective Small Group events is important for developing better connections and friendships. Because of this, we needed to determine the best way to ensure that we placed each student in a Small Group for which they would almost always be available.

In order to accommodate students as well as possible, I attempted to manually assign students to Small Groups at the beginning of the Fall 2019 semester. The idea for this project originated from the desire to reduce the many hours spent developing solutions. For Small Groups, we asked all students to rank their time slot preferences from one to five (with one indicating a strong preference) through a Google Form. They also had the opportunity to select "Not Available" for any number of time slots when they signed up to participate. The list of five time slots was selected before student members had the opportunity to sign up because we chose time slots based on the schedules of non-student members, since they have to travel to campus. The same was true for the student Group Leaders, since they must be present for all Small Group activities throughout the semester. Survey respondents indicated their gender and class year in the sign-up form. As Small Groups are formed, it is important to keep them as balanced as possible in relation to gender and class year so that we can have a diverse mix of individuals, personalities, and people. We aspire to create the most well-rounded spaces that we can. For example, we did not want a person of a certain class year or gender to be in a group by themselves. Due to the demographics of those who signed up, I decided to characterize students as either "Freshman" or "Non-Freshman" to create two categories for this attribute. For gender, "Male" and "Female" were the only two gender options selected by the participants.

Fall 2019 was the first semester for Small Groups, and I assigned the groups by hand. I wondered how I could approach this problem from a mathematical perspective. How could I find an optimal, rather than simply good, solution to allocate students into balanced groups while honoring preferences for time slots? This problem is what led me to Operations Research, a branch of applied mathematics, so I could learn the language behind the mathematical model that I wanted to write for this Villanova Buddies problem.

Data

Table 1 shows the data from the first three students, taken from a sample of the full data table in Appendix 2 (which can be found in the Web version of this article) of the data collected from all of the students that signed up for Small Groups in Spring 2020. Table 2 and Figures 1 and 2 provide summaries of the data collected through surveys.

Table 1. Spring 2020 Small Group Sample Data

Number	Year of School?	Gender	Tuesday Evenings	Wednesday Evenings	Saturday Mornings	Saturday Afternoons	Sunday Afternoons
1	Sophomore	Female	4th Choice	3rd Choice	5th Choice	1st Choice	2nd Choice
2	Junior	Male	Not Available	1st Choice	Not Available	Not Available	Not Available
3	Freshman	Female	2nd Choice	Not Available	Not Available	Not Available	1st Choice

Table 2. Spring 2020 Small Group Participant Composition

Students	Males	Females	Freshmen	Non-Freshman
40	13	27	23	17

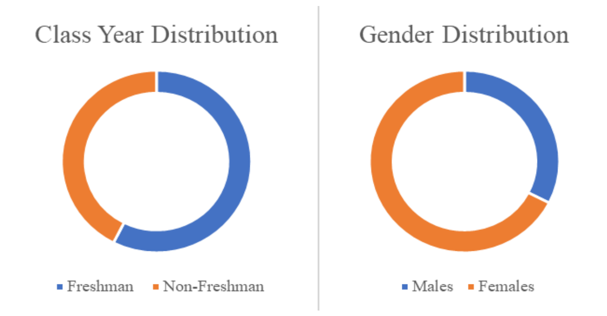


Figure 1. Spring 2020 Small Group Participant Composition

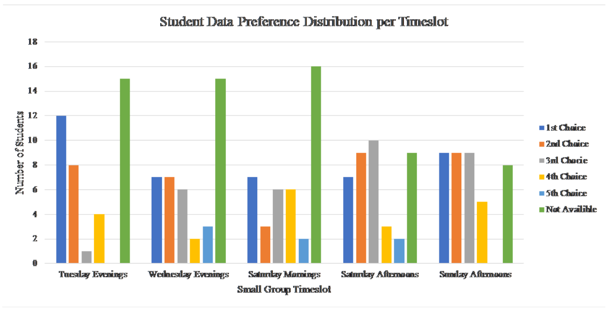


Figure 2. Spring 2020 Small Group Participant Time Slot Preferences

In operations research, optimization problems involve determining the values for decision variables which maximize or minimize an objective function, all while staying within the boundaries of certain restrictions, or constraints. This was an optimization problem because I was attempting to find the best way to do something or best combination of something (1). The decision variables are the student group assignments. Since student scheduling preferences were recorded, with lower values indicating stronger preferences, I wanted to minimize the total of all the students' preferences of all the groups into which they were placed. In order to accomplish this, I summed each student's preference. For example, if Student 1 was placed in their second choice time slot, that added 2 to the total sum of all preferences. The data was converted from text to numbers, as Table 3 displays. In the table that follows, Not Available = 50 because if I were to add all of the numbers and try to find the combination of all numbers to get the lowest value possible, a 50 would never be optimal; it would raise the total value by a substantial amount compared to all other numbers (1-5). Inputting a 50 for "Not Available" prompts the model not to select it because it is much higher than all other values. With the data converted to numbers, I formulated a model to find the best solution. Table 3 shows the data for Spring 2020.

Table 3. Spring 2020 Data

			Preference for each Small Group					Key		
Student	Year	Gender	1	2	3	4	5	Category	Item	Code
1	0	0	4	3	5	1	2		Freshman	1
2	0	1	50	1	50	50	50	Year	Sophomore	0
3	1	0	2	50	50	50	1		Junior	0
4	0	0	50	2	1	3	4		Senior	0
5	1	0	1	50	4	2	3	Gender	Male	1
6	1	0	1	2	50	50	50		Female	0
7	0	1	50	50	2	1	50		1st Choice	1
8	1	0	1	4	3	2	50		2nd Choice	2
9	0	1	4	3	2	1	3		3rd Choice	3
10	1	0	1	3	50	50	2	Preference	4th Choice	4
11	1	0	2	50	4	3	1		5th Choice	5
12	0	1	2	50	50	50	1		Not Available	50
13	0	0	2	3	5	4	1			
14	1	0	50	5	1	3	2			
15	0	0	2	1	50	50	3			
16	0	1	1	50	4	3	2			
17	1	1	1	1	50	3	3			
18	0	0	2	3	50	1	4			
19	0	0	1	2	3	5	4			
20	0	0	4	4	4	4	1			
21	1	1	1	1	50	3	3			
22	1	0	50	50	1	2	3			
23	1	0	50	50	3	1	2			
24	1	0	50	1	3	2	4			
25	0	0	50	50	50	1	2			
26	0	1	50	50	2	1	50			
27	1	1	50	2	4	5	3			
28	1	0	50	2	50	50	3			
29	1	0	50	3	50	4	1			
30	1	0	50	2	50	50	1			
31	0	1	50	50	50	3	1			
32	1	0	1	50	50	3	2			
33	0	1	50	50	1	2	50			
34	1	0	50	5	1	3	2			
35	1	0	50	1	3	2	4			
36	1	0	50	1	4	3	2			
37	0	1	50	50	1	2	3			
38	1	0	50	50	3	2	1			
39	1	0	50	2	50	50	50			
40	1	1	50	50	1	2	50			

Methods and Results

Initial Methodology

The basis of this problem is the understanding that the student is either in or is not in a group (time slot). I defined the variable x_{ij} for each student, where i is the number of the student and j is the group number. The total number of x_{ij} variables is the product of the number of groups and the number of students. In this case, with 40 students and five groups, there were 200 x_{ij} variables. Each student has five variables in this set of students and groups, one for each group. The value of these variables will be either 0 or 1. Thus, I define:

$$x_{ij} = \begin{cases} 1 & if\ student\ i\ is\ in\ group\ j \\ 0 & if\ not \end{cases}$$

for each student. For example, $x_{3,5}$ is the third student's variable for the fifth group, which is Sunday Afternoon. I utilized the preferences for each student in the formulation by multiplying each of the x_{ij} variables with their respective Preference parameter, which I called p_{ij}. For example, $p_{3,5} = 1$ since the Sunday Afternoon time slot was the first choice for the third student. As previously discussed, I wanted to minimize the total sum of all preferences for all students, but only for the groups within which they were placed. I did this by taking the sum of $p_{ij}\ x_{ij}$. Doing this would yield the sum of all the students' preferences only for the groups they were in, because as x_{ij} is defined, it will $= 0$ if the student is not in the respective group, meaning that the whole $p_{ij}\ x_{ij}$ term will $= 0$ as well. The sum is this once formulated: $Z=\Sigma^{40}_{i=1}\Sigma^{5}_{j=1}p_{ij}\ x_{ij}$. This notation takes the sum of the product of x_{ij} and p_{ij} over all the combinations of variables for i=1..40 and j=1..5. This worked because both the x and p variables have the same subscripts, so their respective sizes are the same. Once this sum is taken, the sum is Z.

I used the software package Lingo to generate a solution for all models. Lingo is a product of LINDO Systems that specializes in non-linear, linear, and integer programming (3). This project primarily uses linear programming, with some use of integer programming models (4). At this point, if I ran Lingo with the data and defined these variables in order to minimize Z, I would have received an answer resulting in every student not being placed in a group, which would result in the summation of $p_{ij}\ x_{ij}$ equal to 0. This happens because if a student is not in a group, a 0 is added to the total sum of preferences. Doing this for all students would result in a total of 0, which is the minimum possible value. However, as previously discussed, I had other factors to consider, such as group size, gender, and class year of the students. The most important factor is to formulate the code in such a way that every student must be assigned to a group. Villanova Buddies is focused on inclusion, so it is important that every student is placed in a group. Additionally, students can only be in one group. Writing these two statements together provides the notation that adding up every students' x_{ij} variables should all $= 1$. This means that every student would be in a group, but not more than 1 group. For student 1,

$$x_{1,1} + x_{1,2} + x_{1,3} + x_{1,4} + x_{1,5} = 1$$

the shorter way to write this constraint is $\Sigma^{5}_{j=1}x_{1,j} = 1$. Since this constraint would have to be repeated for all students (i=1..40), $\Sigma^{5}_{j=1}x_{1,j}= 1$ (for i=1..40) is used. This set of constraints assures that every student is included without placing a student in multiple groups.

The next thing I formulated was the group size. I wanted all groups to be the same size, which, in this case, meant eight students per group. The formulation for this constraint is $\Sigma^{40}_{i=1}x_{ij}=8$ (for j=1..5). This summation states that the total of the students for each group (1-5) must be equal to 8. With the x_{ij} variables only $= 1$ when the student is in the group, taking the sum of all the variables for each group time slot overall students, i, will give the number of students that are in each group, j.

Additional considerations included the balance the genders and class years of all the students who signed up for Villanova Buddies. To define this data, it is placed in column vectors aligning with the student numbers, after it is converted to a number, using the following notation:

$$g_i = \begin{cases} 1 & if\ student\ i\ is\ male \\ 0 & if\ not \end{cases}$$

$$y_i = \begin{cases} 1 & if\ student\ i\ is\ a\ freshman \\ 0 & if\ not \end{cases}$$

Each of the two attributes, year and gender, has its own column vector, so each of these are 40 rows by 1 column to match the number of students. The data used after applying the above rules are shown in Table 3. Additionally, g_i and y_i match the rows of the x_{ij} and p_{ij} vectors since they have 40 rows (one for each student). I decided to apply the gender and year constraints by selecting minimum and maximum values for both gender and class year for all groups. Since there are 13 males signed up for Small Groups, that is an average of 2.6 per group. According to previously stated objectives, it was important that no male be the only male in the group, so I set a minimum of 2. A maximum was not needed because with a minimum of 2, no group could have more than 5, which is within the acceptable range for males. However, I decided to apply a maximum of four males per group regardless. By deduction, since there will be eight students per group, the range of females for each group is 4-6. The same logic applies to class year. With close numbers of freshman and non-freshman, 23 and 17 respectively, I decided to place a minimum of 3 and a maximum of 6 freshmen per group. This results in a range of 2-5 non-freshmen per group, which is consistent with the fact that freshmen barely outnumber the non-freshmen. In order to add these limitations to the model, I multiplied g_i and x_{ij} together and considered the sum. Since both of these variables are binary, 1 or 0, the only way to get a non-zero value from $g_i x_{ij}$, is for both of the terms to be 1. This meant that student *i* was placed in group *j* and student *i* is a male. The values that have been determined are the minimum and maximum values of the total males for each group *j*. For the constraint, I wanted to sum the total of $g_i x_{ij}$ for every group and for each of these sums to have minimum and maximum values that I previously determined. From this, the constraints $2 \leq \Sigma^{40}_{i=1} g_i\, x_{ij} \leq 4$ (for $j=1..5$) were formed (4). The same concept used for gender was also applied to class year using its minimum and maximum values that were already determined from the collective numbers of freshmen and non-freshmen signed up. Using this data, the following constraints were formulated as $3 \leq \Sigma^{40}_{i=1} y_i x_{ij} \leq 6$ (for $j=1..5$) (4). Both constraints needed to be repeated for groups 1-5 because all of the Small Groups have the same constraints and gender and year requirements.

Looking at this minimization and all the constraints together, I formed the following model, Model 1, which is called a linear programming model in Operations Research:

$$Min\ Z = \sum_{i=1}^{40} \sum_{j=1}^{5} x_{ij} p_{ij}$$

subject to:

$$\sum_{j=1}^{5} x_{ij} = 1\ (for\ i = 1..40)$$

$$\sum_{i=1}^{40} x_{ij} = 8\ (for\ j = 1..5)$$

$$2 \leq \sum_{i=1}^{40} g_i x_{ij} \leq 4\ (for\ j = 1..5)$$

$$3 \leq \sum_{i=1}^{40} y_i x_{ij} \leq 6\ (for\ j = 1..5)$$

$$x_{ij} \geq 0\ (for\ i\ \ 1..40\ and\ j\ \ 1..5)$$

Appendix 3 shows Model 1 expanded to display all the coefficients plugged in, summations expanded, and constraints required to run the model and obtain an answer.

There are four major assumptions for this model (2). The Certainty assumption is the assumption that all the data used in the model is accurate and known without a doubt. This assumption held strongly for this model because, to best of my knowledge, the data was accurate and comprehensively representative of the students that signed up for Small Groups. The assumption is that all students filled out the form and submitted information that accurately represents their class year, gender, and their preferences and availability. Outside factors such as schedule changes or other commitments that could arise after a student has filled out the form could affect the Certainty of the data.

Proportionality states that the variables are only raised to the first power (2). Additivity states that there are no cross-product terms (2). Proportionality and Additivity both hold perfectly for this problem. No variables are raised to a power other than 1, and none of the variables are multiplied together. Although variables were multiplied by coefficients and vectors of coefficients, they are not multiplied together in this model with other variables.

Divisibility is the assumption that the decision variables (x_{ij}) can be fractions or decimals-- any real number) (2). As written, the model assumes Divisibility. Divisibility would mean that students could be partially in one group and partially in other groups, but this is not what Villanova Buddies wants for the Small Groups. This problem is easier to solve as a linear program rather than an integer problem, so this is what I tried first. The x_{ij} variables were limited to any real number between 0 and 1. The aim was that this would result in only 0 and 1 values, not any fraction or decimal values for the resultant values of the x_{ij} variables. This is a valid approach in linear programming because some problems output integer solutions even when they are not forced to do so (2).

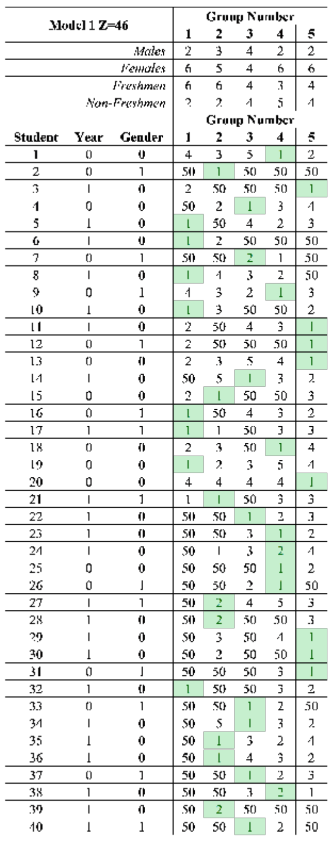
This model is related to the form of an Assignment or Transportation Problem (2) with added constraints. Each student needs to be assigned to one group only. All students must be assigned to a group, and all of the groups must be filled to a certain capacity. The gender and year constraints were added to a traditional Assignment/ Transportation Problem, but this follows the same logic and has the same goal in mind of assigning students to various destinations, which are groups in this problem.

After first formulating this model and understanding the optimization which Lingo would perform, my initial estimate was that the optimal value for Model 1 would be close to $Z=51$. The data showed that there appeared to be a spread of the number of timeslots that students had for their first choice. For example, all respondents did not list one time slot that they all wanted. With this spread, I hypothesized that approximately 25% of the students who signed up would not receive their first choices, and approximately one or two would receive their third choice. This is how I ended up with a value of Z=51, which exceeds 25% more than the minimum optimal value possible, which is Z=40, if everyone receives their first choice.

Initial Results

The solution from Model 1 is shown in Table 4. This table format is the format I will use to present all model data and solutions throughout this report. The highlighted green boxes are the groups in which each student is placed. This format allows us to see which students received which of their time slots and what preference corresponds with the timeslot in which they were placed. The top of this table also provides the statistics needed to compare the groups to one another by the year and gender characteristics. The full Lingo Solution Report can be found in Appendix 4, but all the necessary information from the Report is in Table 4.

Table 4. Model 1 Solution



Model 1 Z=46			Group Number				
			1	2	3	4	5
		Males	2	3	4	2	2
		Females	6	5	4	6	6
		Freshmen	6	6	4	3	4
		Non-Freshmen	2	2	4	5	4
			Group Number				
Student	**Year**	**Gender**	**1**	**2**	**3**	**4**	**5**
1	0	0	4	3	5	1	2
2	0	1	50	1	50	50	50
3	1	0	2	50	50	50	1
4	0	0	50	2	1	3	4
5	1	0	1	50	4	2	3
6	1	0	1	2	50	50	50
7	0	1	50	50	2	1	50
8	1	0	1	4	3	2	50
9	0	1	4	3	2	1	3
10	1	0	1	3	50	50	2
11	1	0	2	50	4	3	1
12	0	1	2	50	50	50	1
13	0	0	2	3	5	4	1
14	1	0	50	5	1	3	2
15	0	0	2	1	50	50	3
16	0	1	1	50	4	3	2
17	1	1	1	1	50	3	3
18	0	0	2	3	50	1	4
19	0	0	1	2	3	5	4
20	0	0	4	4	4	4	1
21	1	1	1	1	50	3	3
22	1	0	50	50	1	2	3
23	1	0	50	50	3	1	2
24	1	0	50	1	3	2	4
25	0	0	50	50	50	1	2
26	0	1	50	50	2	1	50
27	1	1	50	2	4	5	3
28	1	0	50	2	50	50	3
29	1	0	50	3	50	4	1
30	1	0	50	2	50	50	1
31	0	1	50	50	50	3	1
32	1	0	1	50	50	3	2
33	0	1	50	50	1	2	50
34	1	0	50	5	1	3	2
35	1	0	50	1	3	2	4
36	1	0	50	1	4	3	2
37	0	1	50	50	1	2	3
38	1	0	50	50	3	2	1
39	1	0	50	2	50	50	50
40	1	1	50	50	1	2	50

All values fell within the ranges that I defined, and the groups that were formed were balanced in size, gender, and class year. As seen in Appendix 4, the optimal value is found to be Z=46. The lowest possible optimal number was 40 because if every student received the best possible time slot (their first choice), then the sum would simply be 1(1) 40 times over summed together, which would = 40. Since the optimal value is Z=46, this means that six preference points were lost due to one or a combination of the constraints. This could be six students received their second choice, three students getting their third choice, etc., or some combination of these. I ensured that no student was placed in a group for which they are not available. Since the coefficient of p_{ij} is 50 for the "Not Available" selection, if the optimal solution does not exceed 89, then no student could have been placed in a group for which they were not available. The minimum for a student being placed in a group for which they are not available is 89; if all 39 other students received their first choice, the solution would be 1(1) 39 times over and 1(50) 1 time, which is 39+50, or 89.

By analyzing the solution and comparing the group placement to the original data of Availability and Preference, I found that 34 students received their first choice time slots while six were placed in their second choice group. This makes sense because 34(1)+6(2)=46, which verifies the solution above. After looking at the results, the optimal solution of Z=46 matched up with the solution and the initial data. The results showed that the model solves for the optimal groups taking the data into account because all of the students got one of their top two Small Group choices, and the groups are balanced in size, gender, and class year in relation to each other. The optimal preference value of the model is due to the minimization, while the group balance can be attributed to the constraint portion of the model.

Model 1 was used to formulate the groups for the Spring 2020 semester. At the point in the semester when I needed the groups, this was the only model formulated. Using these groups, our attendance went up from an overall attendance rate of 66% in Fall 2019 to 75% in Spring 2020 across all Small Groups: an increase of 9 percentage points. This suggests that students were placed in groups that they wanted, and the groups were balanced, so they were able and willing to attend more than in the past. The groups were quite evenly distributed in class year and gender, which is a desired result of this model. As coordinator of the Small Groups program, I received more positive feedback regarding the groupings for the Spring 2020 semester compared to those of the Fall 2019 semester; the groups that the model produced were more successful than those that I formed without the model. This feedback provides real-world validation to the success of this model. This solution shows that the linear program outputs integer values for the x_{ij} variables. Even though I did not add constraints to force them to be integer values, all the x_{ij} variables were found to be 0 or 1.

This initial solution can be verified by adding the @BIN constraints to all of the decision variables (the x_{ij} variables), which forces them to be 0 or 1. This restriction needed to be satisfied since in real life; we did not want students to be split between groups. The current model does allow that to happen, although it does not in the solution for my data set. This has been checked by adding the @BIN function for all the variables, and the same solution was found as the initial solution. Since this solution matches, the solution is now verified.

Further Development

Model 1 is long and drawn out, with Appendix 3 containing four pages of code, because constraints that were needed for every group or every student were written out for each one. Summations and FOR loops were not used to formulate the model, so the process was long, messy, and not easily applicable for future use. To use this model again, I would have to rewrite every coefficient and add or remove variables depending on the number of students and number of groups. I spent about five hours coding and inputting all of the figures into Model 1. Doing this again would require another two or three, which is not efficient since this process must be done for Small Groups for every semester.

Model 2, along with all the later models of this paper, all call on an Excel file for the data to make it easier to set up. The data that I placed in a file contains the same format and information that comprises the first eight columns of Table 3. This table is used because it has all the data needed for the project in numeric form, all together in one table. To determine the sizes of the Small Groups, this code divides the number of students by the number of groups:

$$\left(\frac{Total\ Students}{Total\ Groups}\right)$$

to determine the average group size. This model uses the formula

$$\left(\frac{Attribute\ Total}{Total\ Groups}\right) \pm 1$$

where attribute corresponds to either the gender or class year numbers. Total gender and total year are the total number of all males and freshmen signed up respectively for Small Groups. This model determines the minimum and maximum values for the gender and year constraints as well as the students per group by using the $\left(\frac{Attribute\ Total}{Total\ Groups}\right)$

calculation, which gives an average of the number of either males or freshmen that each group will have. ± 1 was used for these constraints because if the constraints did not have a $\pm$, then the gender and year constraints for each group would be binding to the average of both males and freshmen per group. The minimum number of males and freshmen for each group is:

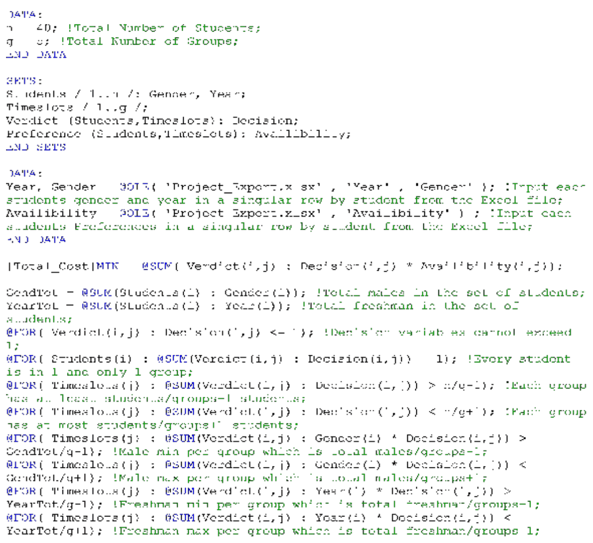
$$\left(\frac{Attribute\ Total}{Total\ Groups}\right)-1$$

while the maximum is

$$\left(\frac{Attribute\ Total}{Total\ Groups}\right)+1.$$

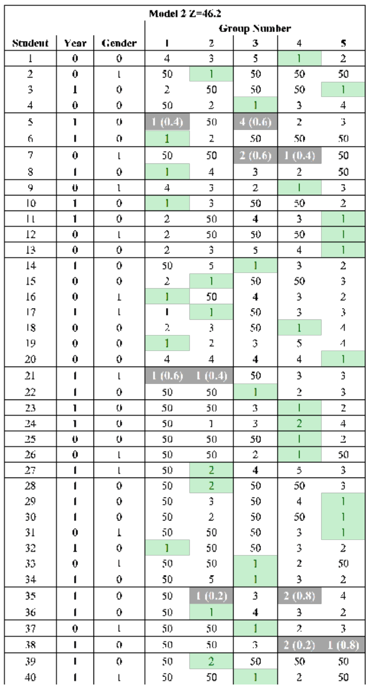
This formulation for the constraints proved to give answers that were not feasible, as the optimal solution shown below was found to be $Z=46.2$ with students 5, 7, 21, 32, and 38 all placed in partial groups. The boxes highlighted in grey correspond to groups that a student is "partially in" according to the code, which is not possible. In the grey boxes, along with the preference for the time slot, I included the decimal value for the solution from Lingo to see how the students were split between groups. Model 2, shown below, was the shortened version of the new model that utilized summations and FOR loops to repeat constraints that are identical. Table 5 shows the solution from this model.

Model 2



```
DATA:
n = 40; !Total Number of Students;
g = 5; !Total Number of Groups;
END DATA

SETS:
Students / 1..n /: Gender, Year;
Timeslots / 1..g /;
Verdict (Students,Timeslots): Decision;
Preference (Students,Timeslots): Availibility;
END SETS

DATA:
Year, Gender = @OLE( 'Project_Export.xlsx' , 'Year' , 'Gender' ); !Input each
students gender and year in a singular row by student from the Excel file;
Availibility = @OLE( 'Project Export.xlsx' , 'Availibility' ) ; !Input each
students Preferences in a singular row by student from the Excel file;
END DATA

[Total_Cost] MIN = @SUM( Verdict(i,j) : Decision(i,j) * Availibility(i,j));

GendTot = @SUM(Students(i) : Gender(i)); !Total males in the set of students;
YearTot = @SUM(Students(i) : Year(i)); !Total freshman in the set of
students;
@FOR( Verdict(i,j) : Decision(i,j) <= 1); !Decision variables cannot exceed
1;
@FOR( Students(i) : @SUM(Verdict(i,j) : Decision(i,j)) = 1); !Every student
is in 1 and only 1 group;
@FOR( Timeslots(j) : @SUM(Verdict(i,j) : Decision(i,j)) > n/g-1); !Each group
has at least students/groups-1 students;
@FOR( Timeslots(j) : @SUM(Verdict(i,j) : Decision(i,j)) < n/g+1); !Each group
has at most students/groups+1 students;
@FOR( Timeslots(j) : @SUM(Verdict(i,j) : Gender(i) * Decision(i,j)) >
GendTot/g-1); !Male min per group which is total males/groups-1;
@FOR( Timeslots(j) : @SUM(Verdict(i,j) : Gender(i) * Decision(i,j)) <
GendTot/g+1); !Male max per group which is total males/groups+1;
@FOR( Timeslots(j) : @SUM(Verdict(i,j) : Year(i) * Decision(i,j)) >
YearTot/g-1); !Freshman min per group which is total freshman/groups-1;
@FOR( Timeslots(j) : @SUM(Verdict(i,j) : Year(i) * Decision(i,j)) <
YearTot/g+1); !Freshman max per group which is total freshman/groups+1;
```

Table 5. Model 2 Solution



Model 2 Z=46.2							
			Group Number				
Student	Year	Gender	1	2	3	4	5
1	0	0	4	3	5	1	2
2	0	1	50	1	50	50	50
3	1	0	2	50	50	50	1
4	0	0	50	2	1	3	4
5	1	0	1 (0.4)	50	4 (0.6)	2	3
6	1	0	1	2	50	50	50
7	0	1	50	50	2 (0.6)	1 (0.4)	50
8	1	0	1	4	3	2	50
9	0	1	4	3	2	1	3
10	1	0	1	3	50	50	2
11	1	0	2	50	4	3	1
12	0	1	2	50	50	50	1
13	0	0	2	3	5	4	1
14	1	0	50	5	1	3	2
15	0	0	2	1	50	50	3
16	0	1	1	50	4	3	2
17	1	1	1	1	50	3	3
18	0	0	2	3	50	1	4
19	0	0	1	2	3	5	4
20	0	0	4	4	4	4	1
21	1	1	1 (0.6)	1 (0.4)	50	3	3
22	1	0	50	50	1	2	3
23	1	0	50	50	3	1	2
24	1	0	50	1	3	2	4
25	0	0	50	50	50	1	2
26	0	1	50	50	2	1	50
27	1	1	50	2	4	5	3
28	1	0	50	2	50	50	3
29	1	0	50	3	50	4	1
30	1	0	50	2	50	50	1
31	0	1	50	50	50	3	1
32	1	0	1	50	50	3	2
33	0	1	50	50	1	2	50
34	1	0	50	5	1	3	2
35	1	0	50	1 (0.2)	3	2 (0.8)	4
36	1	0	50	1	4	3	2
37	0	1	50	50	1	2	3
38	1	0	50	50	3	2 (0.2)	1 (0.8)
39	1	0	50	2	50	50	50
40	1	1	50	50	1	2	50

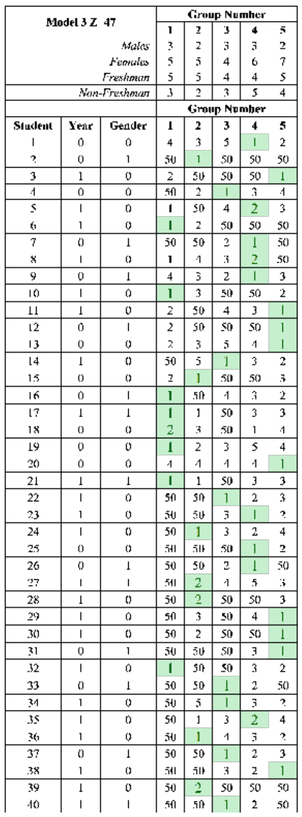
The reason this solution is infeasible is because when (Attribute Total)/(Total Groups)±1 is used, it creates non-integer constants in constraints. For the data used, the total males in each group was 1.6≤*males*≤3.6 and the total freshman per group was defined 3.6≤*freshman*≤5.6. Making these constraints with decimal minimum and maximum values creates an issue that is not present in Model 1. With this change, either the model must have binary variables (integer programming) or the constraint right hand sides must contain whole numbers as they are in Model 1.

The two ways to fix this issue are shown below with two different types of models. Model 3 makes all of the variables binary using the @BIN function in Lingo. This produces an integer optimal solution even though the constraints still have non-whole-number minimums and maximums. The changes made to Model 2 in order to create Model 3 and its resulting solution (Table 6) are both shown below. For this and all later models, model changes take place only in the @FOR lines of Model 2.

Model 3 (changes from Model 2)

```
@FOR( Verdict(i,j) : @BIN(Decision(i,j))); !Decision Variables are binary;
@FOR( Timeslots(j) : @SUM(Verdict(i,j) : Decision(i,j)) > n/g-1); !Each group
has at least students/groups-1 students;
@FOR( Timeslots(j) : @SUM(Verdict(i,j) : Decision(i,j)) < n/g+1); !Each group
has at most students/groups+1 students;
@FOR( Timeslots(j) : @SUM(Verdict(i,j) : Gender(i) * Decision(i,j)) >
GendTot/g-1); !Male min per group which is total males/groups-1;
@FOR( Timeslots(j) : @SUM(Verdict(i,j) : Gender(i) * Decision(i,j)) <
GendTot/g+1); !Male max per group which is total males/groups+1;
@FOR( Timeslots(j) : @SUM(Verdict(i,j) : Year(i) * Decision(i,j)) >
YearTot/g-1); !Freshman min per group which is total freshman/groups-1;
@FOR( Timeslots(j) : @SUM(Verdict(i,j) : Year(i) * Decision(i,j)) <
YearTot/g+1); !Freshman max per group which is total freshman/groups+1;
```

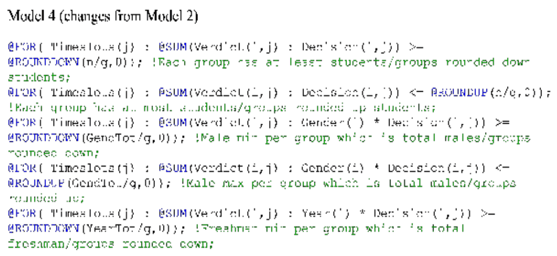
Table 6. Model 3 Solution



Model 3 Z 47			Group Number				
			1	2	3	4	5
		Males	3	2	3	3	2
		Females	5	5	4	6	7
		Freshman	5	5	4	4	5
		Non-Freshman	3	2	3	5	4
			Group Number				
Student	Year	Gender	1	2	3	4	5
1	0	0	4	3	5	1	2
2	0	1	50	1	50	50	50
3	1	0	2	50	50	50	1
4	0	0	50	2	1	3	4
5	1	0	1	50	4	2	3
6	1	0	1	2	50	50	50
7	0	1	50	50	2	1	50
8	1	0	1	4	3	2	50
9	0	1	4	3	2	1	3
10	1	0	1	3	50	50	2
11	1	0	2	50	4	3	1
12	0	1	2	50	50	50	1
13	0	0	2	3	5	4	1
14	1	0	50	5	1	3	2
15	0	0	2	1	50	50	3
16	0	1	1	50	4	3	2
17	1	1	1	1	50	3	3
18	0	0	2	3	50	1	4
19	0	0	1	2	3	5	4
20	0	0	4	4	4	4	1
21	1	1	1	1	50	3	3
22	1	0	50	50	1	2	3
23	1	0	50	50	3	1	2
24	1	0	50	1	3	2	4
25	0	0	50	50	50	1	2
26	0	1	50	50	2	1	50
27	1	1	50	2	4	5	3
28	1	0	50	2	50	50	3
29	1	0	50	3	50	4	1
30	1	0	50	2	50	50	1
31	0	1	50	50	50	3	1
32	1	0	1	50	50	3	2
33	0	1	50	50	1	2	50
34	1	0	50	5	1	3	2
35	1	0	50	1	3	2	4
36	1	0	50	1	4	3	2
37	0	1	50	50	1	2	3
38	1	0	50	50	3	2	1
39	1	0	50	2	50	50	50
40	1	1	50	50	1	2	50

Although the change made to the code by adding the @BIN function solved the problem of having students fractionally divided between groups, a new issue was presented by using this function. Although the code appears to restrict the group sizes to be 7<*groups*<9, but not exactly 7 or 9, this would infer that the group sizes for each timeslot would have to be 8 students. Although this assumption was made, the group sizes in this solution are 7, 8, and 9. After clarification with Lingo syntax, using the < or > signs is the same as using the ≤ or ≥ signs. Therefore, Model 3 actually used 7≤*groups*≤9 for group size. This shows that without using the @ROUND family of functions, there is no clean and easy way to generally write the code to make the group size binding when there is a perfectly divisible number of students per group.

The next modeling approach to this problem is looking at the @ROUND functions (rounding values up or down), and not using the @BIN function. This should resolve both the initial problem of fractional students in groups along with the problem of not having the correct group size that is identified. The next model, Model 4, uses some of the family of @ROUND functions in Lingo. The idea of this model is to use the same thought process discussed earlier, of taking the average students per group along with males and freshmen per group. By doing this, the code calculated the constraint minimum and maximum values itself without having to input them. The key part in this model is that when the average of students, males, or freshmen is used, one of the @ROUND functions is used with it. Using one of the @ROUND functions instructs Lingo to round that number to the nearest specified decimal place. Selecting 0 decimal places will result in a whole number result regardless of whether the average number of students, males, and freshmen is a whole number. In this model, I used @ROUNDUP and @ROUNDDOWN, which instruct Lingo to either round up or round down the value. For the minimum constraints, I used @ROUNDDOWN, and for the maximum I used @ROUNDUP. Using this combination of constraints creates a range into which the values for students, males, and freshmen per group can fall, which is Model 4.

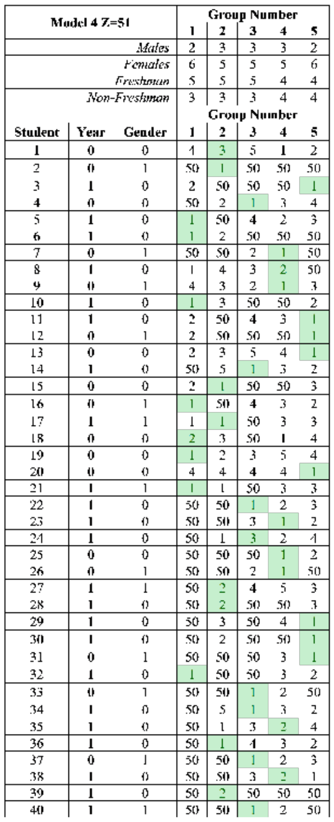

Model 4 (changes from Model 2)

```
@FOR( Timeslots(j) : @SUM(Verdict(i,j) : Decision(i,j)) >=
@ROUNDDOWN(n/g,0)); !Each group has at least students/groups rounded down
students;
@FOR( Timeslots(j) : @SUM(Verdict(i,j) : Decision(i,j)) <= @ROUNDUP(n/g,0));
!Each group has at most students/groups rounded up students;
@FOR( Timeslots(j) : @SUM(Verdict(i,j) : Gender(i) * Decision(i,j)) >=
@ROUNDDOWN(GendTot/g,0)); !Male min per group which is total males/groups
rounded down;
@FOR( Timeslots(j) : @SUM(Verdict(i,j) : Gender(i) * Decision(i,j)) <=
@ROUNDUP(GendTot/g,0)); !Male max per group which is total males/groups
rounded up;
@FOR( Timeslots(j) : @SUM(Verdict(i,j) : Year(i) * Decision(i,j)) >=
@ROUNDDOWN(YearTot/g,0)); !Freshmen min per group which is total
freshman/groups rounded down;
```

For Model 4, sensitivity analysis was performed by changing the ranges for the gender and class year constraints. In Model 4, the ranges for the constraints for both class year and gender are as small as possible at only 1 each, while the group size is held to 8. The @ROUNDUP and @ROUNDDOWN functions in Lingo were used for these constraints because this provides a small range of 1 unit for constraints. The average males per group and freshmen per group are 2.6 and 4.6, respectively. The @ROUNDUP function is used to find the upper bound of these 2 values, which are 3 and 5, by rounding up the two average values (2.6 and 4.6). The @ ROUNDDOWN function is used to find this lower bound of these constraints, which are 2 and 4, by rounding down the two average values (2.6 and 4.6). This is how gender has a range of 2-3 while class year has a range of 4-5. These short ranges restrict the group placements of students and raises the optimal solution to Z=51, which is less favorable than any earlier solution. The complete optimal solution is shown in Table 7.

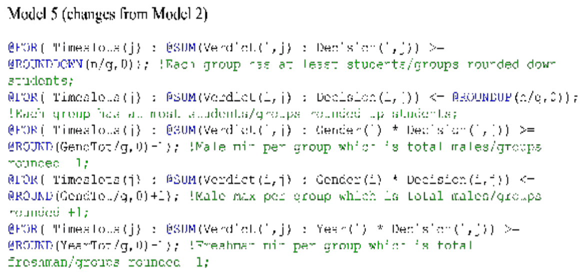
Comparing this to the original model, Model 1, in which the optimal solution was Z=46, more people recieved less desirable preferences than previously. The constraints are tighter, which results in more evenly balanced groups. The next model I evaluated was Model 5, shown below. It uses @ROUND function and then adds or subtracts 1 from that value to get the range for

Table 7. Model 4 Solution



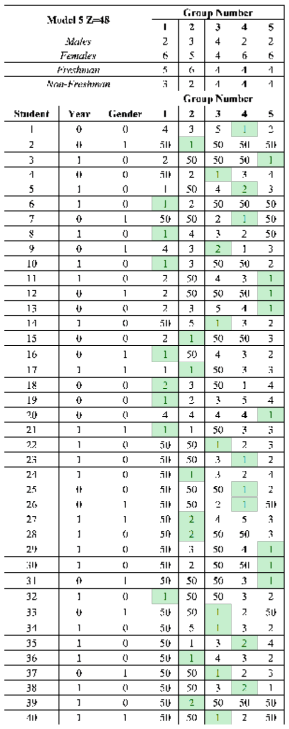
Model 4 Z=51			Group Number				
			1	2	3	4	5
		Males	2	3	3	3	2
		Females	6	5	5	5	6
		Freshman	5	5	5	4	4
		Non-Freshman	3	3	3	4	4
			Group Number				
Student	Year	Gender	1	2	3	4	5
1	0	0	4	**3**	5	1	2
2	0	1	50	**1**	50	50	50
3	1	0	2	50	50	50	**1**
4	0	0	50	2	**1**	3	4
5	1	0	**1**	50	4	2	3
6	1	0	**1**	2	50	50	50
7	0	1	50	50	2	**1**	50
8	1	0	1	4	3	**2**	50
9	0	1	4	3	2	**1**	3
10	1	0	**1**	3	50	50	2
11	1	0	2	50	4	3	**1**
12	0	1	2	50	50	50	**1**
13	0	0	2	3	5	4	**1**
14	1	0	50	5	**1**	3	2
15	0	0	2	**1**	50	50	3
16	0	1	**1**	50	4	3	2
17	1	1	1	**1**	50	3	3
18	0	0	**2**	3	50	1	4
19	0	0	**1**	2	3	5	4
20	0	0	4	4	4	4	**1**
21	1	1	**1**	1	50	3	3
22	1	0	50	50	**1**	2	3
23	1	0	50	50	3	**1**	2
24	1	0	50	1	**3**	2	4
25	0	0	50	50	50	**1**	2
26	0	1	50	50	2	**1**	50
27	1	1	50	**2**	4	5	3
28	1	0	50	**2**	50	50	3
29	1	0	50	3	50	4	**1**
30	1	0	50	2	50	50	**1**
31	0	1	50	50	50	3	**1**
32	1	0	**1**	50	50	3	2
33	0	1	50	50	**1**	2	50
34	1	0	50	5	**1**	3	2
35	1	0	50	1	3	**2**	4
36	1	0	50	**1**	4	3	2
37	0	1	50	50	**1**	2	3
38	1	0	50	50	3	**2**	1
39	1	0	50	**2**	50	50	50
40	1	1	50	50	**1**	2	50

constraints. This combines what I have done in multiple previous models. It uses the logic of utilizing ±1, while also using one of the @ROUND functions so the bounds of the constraints will still be whole numbers. This creates a range from minimum to maximum values for constraints to be 2 units instead of the tighter 1-unit range in Model 4. With the round function being used, 2.6 and 4.6 are rounded up to 3 and 5, respectively. After this, 1 is added or subtracted to this new rounded number to obtain a range for both attributes. Here, gender has a range of 2-4, while class year has the range 4-6. Since these constraints are looser, the optimal value is expected to be lower and this is found to be true. Z=48 for Model 5, and the model and solution (Table 8) are both shown below.


Model 5 (changes from Model 2)

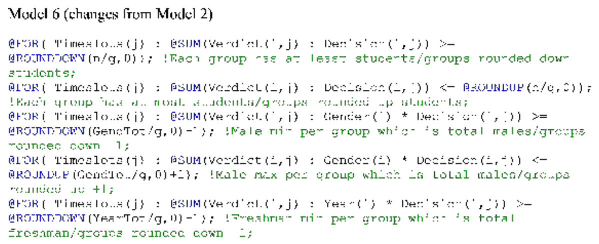
```
@FOR( Timeslots(j) : @SUM(Verdict(i,j) : Decision(i,j)) >=
@ROUNDDOWN(n/g,0)); !Each group has at least students/groups rounded down
students;
@FOR( Timeslots(j) : @SUM(Verdict(i,j) : Decision(i,j)) <= @ROUNDUP(n/g,0));
!Each group has at most students/groups rounded up students;
@FOR( Timeslots(j) : @SUM(Verdict(i,j) : Gender(i) * Decision(i,j)) >=
@ROUND(GendTot/g,0)-1); !Male min per group which is total males/groups
rounded -1;
@FOR( Timeslots(j) : @SUM(Verdict(i,j) : Gender(i) * Decision(i,j)) <=
@ROUND(GendTot/g,0)+1); !Male max per group which is total males/groups
rounded +1;
@FOR( Timeslots(j) : @SUM(Verdict(i,j) : Year(i) * Decision(i,j)) >=
@ROUND(YearTot/g,0)-1); !Freshman min per group which is total
freshman/groups rounded -1;
```

Table 8. Model 5 Solution



Model 5 Z=48			Group Number				
			1	2	3	4	5
	Males		2	3	4	2	2
	Females		6	5	4	6	6
	Freshman		5	6	4	4	4
	Non-Freshman		3	2	4	4	4
			Group Number				
Student	Year	Gender	1	2	3	4	5
1	0	0	4	3	5	1	2
2	0	1	50	1	50	50	50
3	1	0	2	50	50	50	1
4	0	0	50	2	1	3	4
5	1	0	1	50	4	2	3
6	1	0	1	2	50	50	50
7	0	1	50	50	2	1	50
8	1	0	1	4	3	2	50
9	0	1	4	3	2	1	3
10	1	0	1	3	50	50	2
11	1	0	2	50	4	3	1
12	0	1	2	50	50	50	1
13	0	0	2	3	5	4	1
14	1	0	50	5	1	3	2
15	0	0	2	1	50	50	3
16	0	1	1	50	4	3	2
17	1	1	1	1	50	3	3
18	0	0	2	3	50	1	4
19	0	0	1	2	3	5	4
20	0	0	4	4	4	4	1
21	1	1	1	1	50	3	3
22	1	0	50	50	1	2	3
23	1	0	50	50	3	1	2
24	1	0	50	1	3	2	4
25	0	0	50	50	50	1	2
26	0	1	50	50	2	1	50
27	1	1	50	2	4	5	3
28	1	0	50	2	50	50	3
29	1	0	50	3	50	4	1
30	1	0	50	2	50	50	1
31	0	1	50	50	50	3	1
32	1	0	1	50	50	3	2
33	0	1	50	50	1	2	50
34	1	0	50	5	1	3	2
35	1	0	50	1	3	2	4
36	1	0	50	1	4	3	2
37	0	1	50	50	1	2	3
38	1	0	50	50	3	2	1
39	1	0	50	2	50	50	50
40	1	1	50	50	1	2	50

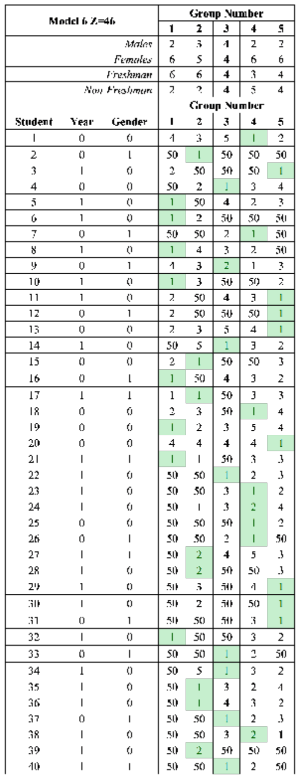
Lastly, the widest range I tested used the @ROUNDUP and @ROUNDDOWN functions and then added or subtracted 1 for Model 6. Similar to Model 4, the @ROUNDUP and @ROUNDDOWN functions are used to find the range of the acceptable attribute values for each group. This model also used ±1 in conjunction with the @ROUNDUP and @ROUNDDOWN functions. The averages of 2.6 and 4.6 are rounded up and down to 2 and 3 for gender and 4 and 5 for freshmen. After this, 1 is subtracted from the minimum, while 1 is added to the maximum. This created a larger range for Model 6 than Models 4 and 5. For Model 6, Z=46, which is the lowest value of the three models analyzed. Its optimal value is identical to Model 1. This comparison makes sense because the ranges for the constraints in the Original Model were 2 and 3 for gender and class year, respectively. In the original model, the constraint bounds were 2≤*males*≤4 and 3≤*freshmen*≤6. The constraints here in Model 6 define the bounds as 1≤*males*≤4 and 3≤*freshman*≤6. These numbers are almost identical, which is why the solutions of this model and the original model would be the same. The model and solution (Table 9) for Model 6 are depicted next.



Model 6 (changes from Model 2)

```
@FOR( Timeslots(j) : @SUM(Verdict(i,j) : Decision(i,j)) >=
@ROUNDDOWN(n/g,0)); !Each group has at least students/groups rounded down
students;
@FOR( Timeslots(j) : @SUM(Verdict(i,j) : Decision(i,j)) <= @ROUNDUP(n/g,0));
!Each group has at most students/groups rounded up students;
@FOR( Timeslots(j) : @SUM(Verdict(i,j) : Gender(i) * Decision(i,j)) >=
@ROUNDDOWN(GendTot/g,0)-1); !Male min per group which is total males/groups
rounded down -1;
@FOR( Timeslots(j) : @SUM(Verdict(i,j) : Gender(i) * Decision(i,j)) <=
@ROUNDUP(GendTot/g,0)+1); !Male max per group which is total males/groups
rounded up +1;
@FOR( Timeslots(j) : @SUM(Verdict(i,j) : Year(i) * Decision(i,j)) >=
@ROUNDDOWN(YearTot/g,0)-1); !Freshmen min per group which is total
freshman/groups rounded down -1;
```

Table 9. Model 6 Solution



Model 6 Z=46			Group Number				
			1	2	3	4	5
		Males	2	3	4	2	2
		Females	6	5	4	6	6
		Freshman	6	6	4	3	4
		Non Freshman	2	2	4	5	4
			Group Number				
Student	Year	Gender	1	2	3	4	5
1	0	0	4	3	5	1	2
2	0	1	50	1	50	50	50
3	1	0	2	50	50	50	1
4	0	0	50	2	1	3	4
5	1	0	1	50	4	2	3
6	1	0	1	2	50	50	50
7	0	1	50	50	2	1	50
8	1	0	1	4	3	2	50
9	0	1	4	3	2	1	3
10	1	0	1	3	50	50	2
11	1	0	2	50	4	3	1
12	0	1	2	50	50	50	1
13	0	0	2	3	5	4	1
14	1	0	50	5	1	3	2
15	0	0	2	1	50	50	3
16	0	1	1	50	4	3	2
17	1	1	1	1	50	3	3
18	0	0	2	3	50	1	4
19	0	0	1	2	3	5	4
20	0	0	4	4	4	4	1
21	1	1	1	1	50	3	3
22	1	0	50	50	1	2	3
23	1	0	50	50	3	1	2
24	1	0	50	1	3	2	4
25	0	0	50	50	50	1	2
26	0	1	50	50	2	1	50
27	1	1	50	2	4	5	3
28	1	0	50	2	50	50	3
29	1	0	50	3	50	4	1
30	1	0	50	2	50	50	1
31	0	1	50	50	50	3	1
32	1	0	1	50	50	3	2
33	0	1	50	50	1	2	50
34	1	0	50	5	1	3	2
35	1	0	50	1	3	2	4
36	1	0	50	1	4	3	2
37	0	1	50	50	1	2	3
38	1	0	50	50	3	2	1
39	1	0	50	2	50	50	50
40	1	1	50	50	1	2	50

Discussion

I attempted to reduce bias by assigning numbers to all of the students instead of keeping names with the calculations and optimization process. Students could have affected their own data submission for availability depending upon friends' selections of time slots. Although this information may not be valid in ranking their availability, it does correctly represent their preference of Small Groups that they want to be placed in, so this does not negatively affect the model.

One way that the model could be stronger would be by having students place a relative weight on their preferences instead of just ranking them 1-5 or "Not Available". Giving one student their second choice versus their first choice might not be important to some people, but it could be important to other people. Lingo considers all these differences equivalent, and no way of measuring these differences has been accounted for in these models. Taking these differences into account would result in a slightly better model of what I am optimizing because it would include the magnitude of difference in desire for group 1 versus group 2 rather than placing the same weighted difference between 1-5 as the current model and survey system does. This change would be easy to make as it would not change the process of finding the minimal optimization, but it would simply require changing the verbiage on the survey and how the data is taken.

In conclusion, the formulation of this model for the Villanova Buddies Small Groups has been a successful tool for creating Small Groups. This model was applied this semester and I will be able to use it in following semesters. In efforts to compare these models to my manual results, I retrofitted Model 3 with handpicked constraint bounds to the Fall 2019 pool of students. By hand, I received a Z value of 64, and these were the assignments actually used for Fall 2019, but using the model, I found a solution that has a Z value of 57. This shows that the model successfully creates significantly better results than fomulating the groups by hand.

Ultimately, Models 4, 5, and 6 are the same Model, but they use different formulas to determine the minimum and maximum values for the constraints. These differences are important because these constraints dictate which students can and cannot be in which groups. Although the differences between the Models are subtle, they can be used together to get a better overall picture of the data present.

When I evaluate another set of students using one of these models, I will use Models 4, 5, and 6 to decide which Model I want to use to place groups. If it is possible to use the results from the Model that uses the tightest constraints, Model 4, without a significant penalty in objective function value compared to Model 6, I will use it. However, there is a chance that Model 4 is not feasible for a data set, or that the value of its objective function (Z) is much higher than Model 6. In the case of the data from the Spring 2020 semester, the difference from Model 4 to Model 6 was 5 Preference units, so I would consider using Model 5 or Model 4 in the future. I believe that it could be worth sacrificing one Preference unit per group in order for all 5 groups to be more properly balanced. Weighing and balancing the differences in total preference points versus the level of balance of gender and class year for all of the groups is key to considering which of the solutions to utilize in the formation of the groups. Considering multiple models helps get a better perspective of the data set that I am working with rather than only looking at one view via a single Model. One of the reasons that I believe that all three models work well for the data is because we have a diverse group of students involved with Villanova Buddies. Because of this, it is easier to divide up all of the groups because there are enough members of every class year and gender to populate the groups.

One thing to keep in mind when comparing all the models is that depending on the data for the students of a given semester, the constraints for the attributes will change. In the Spring 2020 data, it happens that the lower limit on males in a group is 1 in Model 6. However, referring to the idea behind balancing groups in attributes as discussed previously, this would not be an ideal situation. Thus, in the case of placing a male, female, freshman, or non-freshman in a group by him or herself, reverting to the tighter model would be beneficial. In the case of Spring 2020, looking at Model 5 rather than Model 6 would be beneficial because the Model 5 constraints do not allow any student from any attribute to be alone in a group. This is something to keep in mind since these models are formulated for any data set, not just a singular semester's pool of students. This is one of the reasons that a holistic view of all 3 models looking at different constraint ranges provides a balanced view of all the students, making none of the 3 better than the rest, rather than looking at all the solutions in unison to come up with the best fit of groups for a given semester.

There are two ways to assure that through the model, no student will be alone in any group for any attribute. The first way to do this is to manually pick limits and plug them directly into the program by hand. This way is similar to what was done in Model 1, except the same idea can be applied to the later Models 4, 5, and 6 with the Excel shortcut and @FOR loops. Another is by including constraints to make sure that in every group, there are at least two males, two females, two freshmen, and two non-freshmen by adding constraints in all of the models to state this. These constraints would be

$$\sum_{i=1}^{40} g_i x_{ij} \geq 2 \ (for\ j = 1..5) \text{ and } \sum_{i=1}^{40} y_i x_{ij} \geq 2 \ (for\ j = 1..5)$$

to make sure that there are at least two males and two freshmen per group. Making sure that there would be two females and two non-freshmen in each group would utilize the minimum group size, which would be @ROUNDDOWN (*n/g*). After this, 2 would be subtracted to find the maximum for both males and freshmen. If the maximum for males and freshmen is the total of the minimum group -2, then this also means there will be at least two females and two non-freshmen in every group. Therefore, the added constraints to each model would be the following:

$$\sum_{i=1}^{40} g_i x_{ij} \geq 2 \ (for\ j = 1..5)$$

$$\sum_{i=1}^{40} y_i x_{ij} \geq 2 \ (for\ j = 1..5)$$

$$\sum_{i=1}^{40} g_i x_{ij} \leq \left(@ROUNDDOWN\left(\frac{n}{g}\right)\right) - 2 \ (for\ j = 1..5)$$

$$\sum_{i=1}^{40} y_i x_{ij} \leq \left(@ROUNDDOWN\left(\frac{n}{g}\right)\right) - 2 \ (for\ j = 1..5)$$

Going forward, I would like to incorporate these constraints into Models 4, 5, and 6, as this would eliminate the step of making sure that each group would have two students with each attribute. In addition to this, changing the form to sign up for Small Groups to reflect the relative weighted preference rather than the simple ranking 1-5 is another way to improve the models.

ACKNOWLEDGEMENTS

I would like to thank Dr. Peter Muller for connecting me with Dr. Bruce Pollack-Johnson, which got me started on this project. I would like to thank Dr. Pollack-Johnson for mentoring and teaching me the mathematics behind the paper and taking the time to help proofread and edit throughout the process of working on this paper and model.

REFERENCES

1. Andréasson, N., Evgrafov, A., Patriksson, M., & Gustavsson, E. (2020). *An introduction to continuous optimization: Foundations and fundamental algorithms*. Mineola, NY: Dover Publications.
2. Hillier, F. S., & Lieberman, G. J. (2015). *Introduction to operations research*.
3. LINDO Systems (2018). *Lingo Version 18.0*. Chicago, IL.
4. Vanderbei, R. J. (2004). *Linear programming: Foundations and extensions*. Boston, MA: Kluwer.

Author

Joseph Zinno

Joseph Zinno is a member of the Villanova Class of 2022. He is pursuing a Bachelor of Science in Mechanical Engineering with minors in Engineering Entrepreneurship and Mathematics. Joseph is from Steubenville, Ohio and graduated from Catholic Central High School in 2018. His work with Villanova Buddies, a student organization, led him to take Operations Research to delve deeper into constructing a mathematical model for formulating Small Groups, with the goal of developing a usable model for the organization. Joseph plans to work in the product design and development field after graduation.

Mentor

Dr. Bruce Pollack-Johnson

Bruce Pollack-Johnson earned his B.A. in Sociology with a minor in Education from Brandeis University, his M.A. in Applied Mathematics from Temple University, and M.S. and Ph.D. in Operations Research from the University of Pennsylvania Wharton School. His dissertation centered on optimal hiring policies for universities when demand is cyclical. He has published dozens of articles on forecasting, project scheduling, educational modeling, analytics, pedagogy, and statistics. Together with Audrey Borchardt, he wrote three editions of a two-volume text on Business Calculus focusing on modeling and everyday life. A faculty member in the Department of Mathematics & Statistics at Villanova since 1990, he has initiated courses on Math and Fairness and Math and Social Justice.