Classroom Assignment to Minimize Vacancy in Classes

Bryan Ding, Sahil Abraham, Jordan Klodowski, Foster Boales

December 17, 2018

1 Introduction

A common dilemma for Carnegie Mellon students when registering for classes is the inevitability of facing long waitlists. For both the students and the departments, it becomes an unpleasant scenario when classrooms are assigned suboptimally to the demand met by the students. For instance, it would be unwise to have a classroom assignment where the capacity of the room is far smaller than the amount of students wishing to take the course, and likewise, it would also be unwise to have a classroom where the amount of empty seats greatly exceed the number of students registered for the course. Our goal is to find a way to minimize waitlist lengths for classes, by optimizing the amount of students placed into a room for a course, and by minimizing the number of classes with multiple empty seats. If we can use large and small classrooms optimally so that we eliminate as many empty seats as possible while still getting everyone into their desired courses, we could potentially minimize the number of students on waitlists across campus by a considerable degree. We will accomplish this by assigning classes to rooms based on room capacity, time availability for each classroom, and student demand for each individual course.

2 Model Assumptions

To approach this problem, there are several important assumptions we need to make. Although not all are realistic, the assumptions provide us a reasonable framework to work under. The assumptions are as follows:

1. Each class can have more than 1 lecture, but we disregard recitations.

2. Departments do not hold priorities over specific rooms.

3. Walking distance is negligible. That is, we do not need to worry about walking across buildings while deciding an assignment.

4. Time is measured from 0 to 13, where 0 represents 8:30 AM to 9:30 AM and 13 represents 9:30 PM to 10:30 PM.

5. Every class is only one hour to make up for the fact that some classes are 50 minutes, which is slightly less than an hour, and some classes are 80 minutes, which is slightly over an hour.

These assumptions could be loosely modified to adapt to different situations, but for now, this is a framework we will use to construct a simple, yet informative, model.

3 Objective Function

Our problem is essentially an integer linear programming problem. The variables that we need to have are

- Course x.
- Classroom k.
- Capacity of classroom k: A_k.
- Course capacity of course x: C_x.

Furthermore, we have the following variable to account for courses taking place in a certain classroom:

$$y_{k,x} = \begin{cases} 1 & \text{if course } x \text{ happens in room } k, \\ 0 & \text{else.} \end{cases}$$

Our problem is thus:

$$\begin{aligned} \text{Minimize} \qquad & \sum_k \sum_x y_{k,x}(A_k - C_x) \\ \text{subjected to} \qquad & \sum_k y_{k,x} = 1 \qquad \text{for all } x \\ & C_x y_{k,x} \leq A_k y_{k,x} \qquad \text{for all } x, k, \\ & \sum_x y_{k,x} \leq 13 \qquad \text{for all } k. \end{aligned}$$

It is important to note that the time constraint t is not directly inputted into the formulation. This is because we can solve this problem for one particular time and then apply it generally. Also note that we do not have a variable for students, because the actual student is not important in the formulation; we only care about the classroom size and the course capacity.

4 Explaining the Constraints

First, observe what we are trying to minimize. We want to minimize

$$\sum_k \sum_x y_{k,x}(A_k - C_x),$$

because this expression ensures that we are subtracting the capacity of a course from its classroom capacity. This is multiplied by the indicator variable $y_{k,x}$ because we do not want to consider courses that do not occur in rooms at a certain time. By summing over k and x, we efficiently manage to consider all possible combinations of courses and classrooms for a particular time slot.

The first constraint we have is that

$$\sum_k y_{k,x} = 1 \qquad \text{for all } x.$$

This constraint ensures that we can match a course with an appropriate classroom at a given time.

The second constraint we have is that

$$C_x y_{k,x} \leq A_k y_{k,x} \qquad \text{for all } x, k.$$

This is necessary because we want to ensure that the actual course capacity of a certain class does not exceed the actual classroom capacity.

The final constraint we have is that

$$\sum_x y_{k,x} \leq 13 \quad \text{for all } x.$$

This is necessary to ensure that the total number of courses in one day does not exceed 13, which is the same as the number of hours in one academic day. (We assume that $0 \leq t \leq 13$, where $t = 0$ corresponds to 8:30 AM to 9:30 AM and $t = 13$ corresponds to 9:30 PM to 10:30 PM).

5 Solving the Problem

We will solve the problem of assigning Spring 2019 math courses. To solve this problem, we take two approaches. For the first approach, we can use the Greedy Algorithm to assign courses to classrooms with the closest capacities. However, in the Greedy Algorithm we will allow rooms to be assigned multiple times to different courses. The goal of this is so we can see that *if* we had access to any classroom at any time, which classroom would be best suited to which class. Of course, this by itself is not enough, because in real life, courses are assigned at different times. We will then complement this by applying the Hungarian Algorithm on subsets of the data with their respective matchings obtained from the Greedy Algorithm to assign a maximum of ten courses with each time.

5.1 Greedy Algorithm

To apply the Greedy Algorithm, we rank all the classroom and course capacities and start assigning them based on how well their capacities match. We will do this for the Department of Mathematical Sciences as an example, and we will try to impose the following constraints in our Greedy Algorithm:

1. To reduce walking distance as much as possible, we will try to assign math courses to buildings as close to Wean Hall as possible, if not in Wean Hall itself. The actual student schedule will vary based on their choice of courses, but by assigning as many math courses in Wean Hall as possible, we can reduce congestion in the center of the campus when students are moving between classes. We rank the order of the buildings based on their approximity to Wean Hall as follows:

   (a) Wean Hall.

   (b) Doherty Hall.

   (c) Porter Hall.

   (d) Baker Hall.

   (e) Hamerschlag Hall.

   (f) Gates-Hillman Center.

   (g) Posner Hall.

(h) Margaret Morrison Hall.

(i) Scaife Hall.

It is important to note that in our assignment, we prioritize matching classrooms with courses with similar capacity sizes over the actual location of the classroom.

2. We will also forsake recitations attached to math courses and focus only on the lectures.

3. For realistic purposes, we will try to limit our time from 8:30 AM to 4:30 PM. We can have multiple courses at the same time, but likewise so can other departments, so we will make the assumption that we cannot have more than 10 courses at any given time.

Using Excel, we can rank classrooms by their capacities. Figure 1 below shows the top 20 classrooms/lecture halls with the most capacities.

	A	B
1	**Classroom**	**Capacity**
2	Doherty 2210	289
3	Doherty 2315	266
4	Gates 4401	244
5	Porter 100	217
6	Posner 160	180
7	Wean 7500	150
8	Baker A51	144
9	Posner A35	144
10	Doherty A302	132
11	Doherty 2302	107
12	Hamerschlag B103	101
13	Scaife 125	99
14	Hamerschlag B131	98
15	Doherty 1212	96
16	MM A14	96
17	Baker 136A	94
18	MM 103	94
19	Posner 152	86
20	Posner 153	86
21	Posner 151	76

Figure 1: The top 20 classrooms/lecture halls with the largest capacities.

Figure 2 below shows the top 20 math courses with the largest capacities:

Course	Capacity
21-270	160
21-122B	152
21-122A	150
21-127A	120
21-127B	120
21-127C	120
21-256A	120
21-256B	120
21-259A	120
21-260A	120
21-240	90
21-241B	90
21-241C	90
21-259B	90
21-260B	90
21-228	86
21-484	78
21-292	73
21-301A	70
21-112	60

Figure 2: The top 20 math courses with the largest capacities.

We now apply the Greedy Algorithm. One particular matching is given as follows:

Course	Capacity	Match	
21-270	160	Posner 160	
21-122B	152	Posner 160	
21-122A	150	Wean 7500	Perfect
21-127A	120	Doherty A302	
21-127B	120	Doherty A302	
21-127C	120	Doherty A302	
21-256A	120	Doherty A302	
21-256B	120	Doherty A302	
21-259A	120	Doherty A302	
21-260A	120	Doherty A302	
21-240	90	Baker 136A	
21-241B	90	Baker 136A	
21-241C	90	Baker 136A	
21-259B	90	Baker 136A	
21-260B	90	Baker 136A	
21-228	86	Posner 152	Perfect
21-484	78	Posner 153	
21-292	73	Baker A53	Perfect
21-301A	70	Baker A53	
21-112	60	Baker A53	
21-120	60	Baker A53	
21-124	60	Baker A53	
21-261	60	Baker A53	
21-268	60	Baker A53	
21-269	60	Baker A53	
21-241A	48	Wean 5403	Perfect
21-420	45	Wean 4623	Perfect
21-499	40	Wean 5302	Perfect
21-301B	35	Baker 235A	Perfect
21-325	35	Baker 235B	Perfect

21-329	35	Baker 237B	Perfect
21-341	35	Baker 255A	Perfect
21-344	35	Baker 235A	Perfect
21-355A	35	Baker 235B	Perfect
21-355B	35	Baker 237B	Perfect
21-356	35	Baker 255A	Perfect
21-373	35	Baker 235A	Perfect
21-101	30	Wean 4709	Perfect
21-242	30	Wean 5310	Perfect
21-369	30	Wean 5312	Perfect
21-111	25	Wean 8427	Perfect
21-236	24	Wean 5328	Perfect
21-238	24	Wean 6423	Perfect
21-374	20	Wean 5304	Perfect
21-604	20	Wean 5316	Perfect
21-623	20	Porter A19	Perfect
21-640	20	Porter 125D	Perfect
21-660	20	Wean 5304	Perfect
21-702	20	Wean 5316	Perfect
21-703	20	Porter A19	Perfect
21-721	20	Porter 125D	Perfect
21-723	20	Wean 5304	Perfect
21-737	20	Wean 5316	Perfect
21-765	20	Porter A19	Perfect
21-801	20	Porter 125D	Perfect
21-820	20	Wean 5304	Perfect
21-832	20	Wean 5316	Perfect
21-882	20	Porter A19	Perfect
21-272	18	Porter A19C	Perfect
21-606	15	Baker A54	Perfect
21-630	15	Baker A54	Perfect
21-400	12	Porter A20	Perfect

Figure 3: Courses matched with classrooms with closest capacities. The fourth column indicates if the classroom perfectly matches the capacities needed for the course.

There are a few interesting remarks about the Greedy Algorithm and its solution. First, with the Greedy Algorithm, there is no need for *backtracking*. The moment we find an applicable room, we immediately assign it. We see that the course with the highest capacity is 21-270, and the room with the closest size capacity is Posner 160. This by default means that we do not consider any classroom/lecture hall that has a higher capacity than that of Posner 160. We also notice that there are many rooms repeated. This comes naturally from the fact that many courses might have set course capacities, and as such, there are specific rooms that specifically suit those capacities. To offset the fact that many of these rooms are shared across different courses, it becomes important to allocate a time for each of them such that none of them overlap. To do that, we will implement the Hungarian Algorithm to allocate the classrooms with their respective courses.

5.2 The Hungarian Algorithm

The Hungarian Algorithm is a particular method for solving assignment problems. We have found which courses are best associated with which classrooms. The question now is to assign a subset of these courses with times such that there is no overlap between classrooms. We make the following assumptions about our model:

1. Classes with multiple lectures will not take place during the same time. This is a realistic assumption to make scheduling-wise if we want students to have more options in terms of scheduling their classes. (Hence, for instance, 21-122 has two lectures, but we will not include both of them during the same hour).

2. We have a maximum of ten math courses at any given time, with 5 of them at most being lecture halls (i.e. at least 90 students in the classroom). This is another realistic assumption to make since we don't want to put too many courses that require lecture halls at once.

Let us consider a subset of the courses in the math department for our first iteration of the Hungarian Algorithm. Suppose we consider

$$\{\text{21-270, 21-122 A, 21-127 A, 21-256 A, 21-259 A 21-325, 21-329, 21-341, 21-344, 21-355 A}\}$$

for time $t = 1$. We will also consider an arbitrary subset of the classrooms that we deem are "reasonable" for this subset of classes:

$$\{\text{POS 160, WEH 7500, BH A51, DH A302, POS A35}\}\cup$$
$$\{\text{BH 235A, BH 235B, BH 237B, BH 255A, SH 220}\}.$$

Then by applying the Hungarian Algorithm, our matrix for the 5 lecture halls would look like

$$\begin{pmatrix} 20 & 0 & 0 & 0 & 0 \\ 30 & 0 & 0 & 0 & 0 \\ 60 & 30 & 24 & 12 & 24 \\ 60 & 30 & 24 & 12 & 24 \\ 60 & 30 & 24 & 12 & 24 \end{pmatrix}.$$

The rows are the courses 21-270, 21-122 A, 21-127 A, 21-256 A, and 21-259 A respectively, and the columns are the classrooms POS 160, WEH 7500, BH A51, DH A302, and POS A35, respectively.

By applying the algorithm, we get that the optimal assignment at time 1 is

$$\begin{pmatrix} \mathbf{20} & 0 & 0 & 0 & 0 \\ 30 & \mathbf{0} & 0 & 0 & 0 \\ 60 & 30 & \mathbf{24} & 12 & 24 \\ 60 & 30 & 24 & \mathbf{12} & 24 \\ 60 & 30 & 24 & 12 & \mathbf{24} \end{pmatrix},$$

where the numbers bolded are the rooms selected. Hence, we would assign 21-270 to POS 160, 21-122 A to WEH 7500, 21-256 A to BH A51, and 21-259 A to POS A35.

To assign the non-lecture hall classes we follow a near identical procedure. Our matrix for the 5 classes 21-325, 21-329, 21-341, 21-344, and 21-355 A alongside the rooms BH 235A, BH 235B, BH 237B, BH 255A, SH 220 would be

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

Notice that this solution appears to be "trivial" since all five classrooms match the course capacities perfectly. Hence, it becomes obvious that a solution is

$$\begin{pmatrix} \mathbf{0} & 0 & 0 & 0 & 0 \\ 0 & \mathbf{0} & 0 & 0 & 0 \\ 0 & 0 & \mathbf{0} & 0 & 0 \\ 0 & 0 & 0 & \mathbf{0} & 0 \\ 0 & 0 & 0 & 0 & \mathbf{0} \end{pmatrix}.$$

We can repeat this for every instance by selecting up to 5 lecture halls arbitrarily and up to 5 smaller courses. We then proceed by picking reasonable classrooms for them to be assigned to by the Hungarian Algorithm. Our results are:

Time 0 Classes	**Room**	**Wastage**		**Time 3**	**Room**	**Wastage**
21-270	POS 160	20		21-241 C	BH 136A	4
21-122 A	WEH 7500	0		21-484	POS 152	8
21-127 A	BH A51	24		21-292	POS 151	3
21-256 A	DH A302	12		21-301 A	BH A53	3
21-259 A	POS A35	24		21-236	WEH 5328	0
21-325	BH 235A	0		21-374	WEH 5304	0
21-329	BH 235B	0		21-604	WEH 5316	0
21-341	BH 237B	0		21-623	PH A19	0
21-344	BH 255A	0		21-630	BH A54	0
21-355 A	SH 220	0		21-400	PH A20	0
Time 1				**Time 4**		
21-122 B	POS 160	28		21-268	GHC 4307	15
21-127 B	BH A51	24		21-269	BH A53	13
21-256 B	POS A35	24		21-241A	WEH 5403	0
21-260 A	DH A302	12		21-238	WEH 5328	0
21-240	MM 103	4		21-640	WEH 5304	0
21-301B	BH 235A	0		21-660	WEH 5316	0
21-355B	BH 235B	0				
21-356	BH 255A	0		**Time 5**		
21-373	SH 220	0		21-112	BH A53	13
21-499	WEH 5302	0		21-120	GHC 4307	15
				21-702	WEH 5304	0
Time 2				21-703	WEH 5316	0
21-127 C	DH A302	12		21-721	PH A19	0
21-241 B	BH 136A	4				
21-259 B	MM 103	4		**Time 6**		
21-260 B	MM A14	6		21-124	BH A53	13
21-228	POS 152	0		21-261	GHC 4307	15
21-420	WEH 4623	0		21-723	WEH 5304	0
21-101	WEH 4709	0		21-737	WEH 5316	0
21-242	WEH 5310	0		21-765	PH A19	0
21-369	WEH 5312	0		21-606	BH A54	0
21-111	WEH 8427	0				
				Time 7		
				21-801	WEH 5304	0
				21-820	WEH 5316	0
				21-832	PH A19	0
				21-882	PH 125D	0
				21-272	PH A19C	0

Figure 4: Results from the Hungarian Algorithm.

6 Conclusions

There are several conclusions that we can draw from this. First, we notice that in the math department, there is no need for any course to assign a classroom larger than Posner 160 (i.e. there is no need to assign Doherty 2210, Doherty 2315, Gates 4401, or Porter 100). Of course, this seems a bit arbitrary because we need to have data on other departments and how they would like to assign classes. Another important conclusion is that we do not necessarily need to have 10 courses at every time. In fact, doing that would lead to a suboptimal assignment of classrooms because many courses, especially the graduate-level ones, have similar capacities so we would prefer spreading these courses across different times rather than having them all at once. Finally, a very important conclusion is that our Greedy Algorithm was quite efficient in matching courses with classrooms. We saw above that courses assigned with the Greedy Algorithm led to a lot of perfect matches, especially for courses with capacities fewer than 48. It is thus not surprising that when we applied the Hungarian Algorithm to these courses, a lot of wastage is actually 0. This hints that this is indeed an optimal assignment.

7 Future Studies

Of course, we have only applied this algorithm to one department within Carnegie Mellon University. For this to be more realistic, we would have to apply this to every single department by looking at every single course and every single possible time. Doing this will require more work, but given that we have completed a draft of our project with one department, the next step would be to expand our course choices. It is also important to note that our Hungarian Algorithm was rather arbitrary in terms of courses selected. This can also be perfected once we have more courses across departments so we can better assign our classrooms. Lastly, there are other rooms that are not being considered in this project, namely ones that are reserved by the math department such as Wean 7201, Wean 8201, and Wean 8220. These rooms are often used by the math department, but since they are reserved *only* for the math department, we might not be able to include them in a more generalized implementation of this project.

8 References

https://www.cmu.edu/es/docs/classrooms.pdf
https://enr-apps.as.cmu.edu/open/SOC/SOCServlet
http://www.hungarianalgorithm.com/index.php