Dynamic Hungarian Algorithm for CMU Classroom Assignment

Gean Shen, Ziniu Wu, Qiuyu Wang and Hanying Xu

***Abstract*— The classroom assignment of Carnegie Mellon University (CMU) is mainly about optimizing the use of all classrooms for different classes in CMU. The goal of our project is to improve current classroom assignment of CMU and to make a better utilization of different classrooms in order to provide greatest convenience for students. We mainly focused on leaving reasonably space for different classes for both the students who are already in class and for students who are on the waiting list and want to go to class to learn the materials. Another thing that we want to improve on the base of the current system is to minimize the walking distance for students. We are going achieve this by putting the classes of the same college in fewest buildings as possible. The main algorithm we use for our project is doing Hungarian algorithm on all the data we collected from CMU website which includes the Course Number, Course Title, Section, Days, Time Slot, Room and Class Capacity. The main programming languages we used are R and Python.**

I. Background

Right now, the classroom assignment at CMU seems pretty reasonable. It managed to have rooms for all the classes at CMU. However, there are many points that can be improved. Firstly, we find that a lot of classes even though have enough room for current registered students, they do not have enough room for students who are on the waiting list who also want to go to classes and listen materials. On the other hand, we observe that there are a lot of classrooms which is too big for the current registered class. For example, for our 21-393 *Operations Research II* classroom which is pretty big and nice, but there is not class in there from 12:30-1:30 which is a big waste. And for 15-122 *Principles of Imperative Computation* class this year, the classroom is not big enough for all the students. We found that there are a lot of students who need to sit on stairs. Another example is 15-150 *Principles of Functional Programming* recitation, there is not enough computers for each students while another bigger cluster is empty. We really want to improve the system right now in order to fix this problem and leave reasonable empty space for classes in order to let students who are on the waitlist to go to classes as well. Besides this problem, another thing we find is that different classes from the same college are actually located in a lot of different buildings. For example, I used to have a math class is in Scaife Hall and another math class right after it in Doherty Hall. These two buildings are pretty far away, I always did not get enough time to walk between classes. This causes a lot of inconvenience for students especially in winter when the weather is very bad. The problem of not having enough time to walk between classes causes students being late for classes and missing some important pieces of information at the beginning of the classes. In our project, we are also trying to solve this problem as well.

*Math department, Carnegie Mellon University

II. Technical Assumptions

1) During the weekdays, we assume that lectures of a certain course will take place in the same classroom. For example, this course, 21-393 *Operations Research II*, will take place in the same classroom on Monday, Wednesday and Friday across the semester.
2) During the weekdays, we assume that recitations of a certain course will be considered and assigned separately from their corresponding lectures. For example, *Physics I for Engineering Students*, will have recitations on Tuesday and Thursday, which will take place in completely different classrooms from their lecture classroom. The first assumption applies to all recitations as well.

3) During the weekdays, at most one course can take place in a certain classroom.
4) During the weekends, no lecture or recitation will take place.
5) In an effort to simplify the assignment problem and exclude courses with no classroom required, we selected the following departments to study: History, Psychology, Civil Engineering, Material Engineering, Economics, Social Decision Science, Biology, Physics, Computational Biology, Language Technology Institute, Bio-medical Engineering, Electrical and Computer Engineering, Mechanical Engineering, English, Modern Language, Statistics, Chemistry, Computer Science, Machine Learning, Chemical Engineering, Engineering & Public Policy, Robotics, Information Systems, Philosophy, Mathematics, Human-Computer Interaction Institute, Business Administration and Institute for Politics.
6) No course will have specific need towards technical equipment, that is, any classroom can be a qualified candidate for a certain course as long as the capacity requirement is satisfied.

III. Data Collection

Since we are optimizing the classroom assignment for a number of courses, we selected courses from the **spring semester of 2019** for this purpose.

For the classrooms, we collected the location, room number and capacity from the CMU reservation system. If a classroom happens to have no specific capacity, we took the maximum of capacity of the courses taking place at this classroom before.

For the courses, we collected the course number, title, section, days, time slot, maximum enroll, currently assigned classroom from spring 2019 in Schedule of Classes and Student Information Online (SIO) at CMU.

Overall, there are 18 buildings, 178 classrooms and 1871 courses (lectures and recitations).

Room	Capacity	Room	Capacity
BH 136A	111	MI 409	40
BH 136E	20	MI 448	25
BH 140C	20	MI 622	20
BH 140CE	50	MI 838	20
BH 140CF	50	MM 103	96
BH 140E	30	MM 17D	18
BH 140F	30	MM C4	15
BH 145C	18	NSH 1305	70
BH 150	15	NSH 3002	46
BH 154A	12	NSH 3206	25
BH 229A	6	PH 100	217
BH 235A	35	PH 107E	60
BH 235B	35	PH 107H	6
BH 237B	35	PH 125B	26
BH 255A	35	PH 125C	24
BH 255B	18	PH 125D	20
BH 336A	18	PH 126A	24
BH 336B	20	PH 223D	25
BH 340A	20	PH 225B	26
BH 342F	18	PH 226A	26
BH A51	144	PH 226B	28
BH A53	73	PH 226C	28
BH A54	15	PH 27	20
BH A60N	20	PH 7F	50
CFA 303	20	PH A17	18
CFA 317	8	PH A18A	40
CIC 1201	25	PH A18B	38
CYH 100A	20	PH A18C	40
DH 1117	30	PH A19	20
DH 1209	24	PH A19A	20
DH 1211	48	PH A19C	18
!DH 1212	96	PH A19D	18
DH 1302	35	PH A20	12
DH 2105	30	PH A20A	21
DH 2122	30	PH A21	21
DH 2210	289	PH A21A	21
DH 2302	113	PH A22	30
DH 2303	30	PH A7C	27
DH 2315	266	POS 146	50
DH 3200	5	POS 147	22
DH 3207	15	POS 151	60
DH 3302	120	POS 152	86
DH 4201	30	POS 153	86
DH 4303	12	POS 160	180
DH A100	50	POS A35	143
DH A200	40	SH 125	96
DH A301D	45	SH 208	30
DH A302	120	SH 212	20
DH A303	60	SH 214	45
DH A324	30	SH 219	45
DH A325	25	SH 220	35
DH A331	40	SH 222	35
DH C302	50	TEP 2610	42
DH MA341	16	TEP 2611	75
EDS 125	24	TEP 2612	45
GHC 4101	26	TEP 2613	25
GHC 4102	40	TEP 2700	80
GHC 4211	42	TEP 2701	42
GHC 4215	58	TEP 2702	24
GHC 4301	28	TEP 3801	70
GHC 4303	55	TEP 3808	45
GHC 4307	75	WEH 3701	20
GHC 4401	244	WEH 4623	30
GHC 5207	20	WEH 4625	38

Room	Capacity	Room	Capacity
GHC 5222	38	WEH 4709	30
GHC CLSTR	30	WEH 5201	30
HBH 1002	64	WEH 5202	45
HBH 1204	60	WEH 5207	20
HBH 2003	35	WEH 5302	35
HH 1107	8	WEH 5304	20
HH 1305	30	WEH 5310	30
HH A101	30	WEH 5312	30
HH A104	28	WEH 5316	20
HH B103	101	WEH 5320	30
HH B131	98	WEH 5328	24
HH C105	30	WEH 5336	25
HH C110	8	WEH 5403	48
HH A301	150	WEH 5409	48
HL 106B	50	WEH 5415	45
HL 106C	35	WEH 5421	45
HL A10	25	WEH 6327	10
HL A5	20	WEH 6423	26
HL CLSTR	50	WEH 7201	20
MI 301B	10	WEH 7218	20
MI 304	40	WEH 7316	40
MI Auditorium	348	WEH 7423	25
MI 348	80	WEH 7500	152
MI 355	25	WEH 8201	20
MI 357	60	WEH 8427	25

IV. Methodology

Now we got a whole list of courses information, including course numbers, course names, start time, ending time, max enrollment, and all classroom information, including classroom name, buildings and capacity. Before proceeding to develop our algorithm, we needed to process data first.

In CMU, courses take place from Monday to Friday and from 8:30 in the morning to 9:30 at night. We first optimized the day in a week which has the most courses. With that day, we divided the day into 26 time slots, with each one has $0.5hr$ time interval. We then optimized the time interval which has the most courses.

According to out data set, we found out Wednesday has the most number of courses, and within Wednesday, the time interval from 1:30 pm to2:00 pm has the most number of courses than other time interval. Then we mainly performed Baseline Hungarian Algorithm, Baseline with relaxation, and Dynamic Hungarian Algorithm on the specific interval, as discussed in later parts. We then assigned the classrooms for these courses in other time intervals. For example, our course *21-393 Operations Research II* starts from 1:30 pm and ends at 2:30 pm, if we completed the assignment problem for time slot from2:00 pm to 2:30 pm, the classroom for *21-393 Operations Research II* is no longer available for other courses. The same thing applies to courses starting from 1:30 pm to 2:50 pm as well. We assign the classroom directly for these courses taken place in other time slots.

For courses in time slots which have not been optimized, we sort them again, and perform the algorithm on the time slot with the next most courses. We repeat the procedure until all courses have been assigned.

A. Baseline Hungarian Algorithm

The Baseline Hungarian Algorithm only tries to minimize the empty seats for each class. The algorithm takes in two inputs, a vector X of length n indicating the capacity of n class, and a vector Y of length m indicating the capacity of m classes.

Since we want to minimize the empty seats of each class, we then set every entry in the cost matrix to be the difference between classroom capacity and class max enrollment. If the difference is negative, in the sense that the classroom does not have enough capacity for that course, we set the entry to be infinity in order to avoid the situation that there is no enough seats for a class. The pseudo code is listed below titled $Algorithm1HA$.

Algorithm 1 HA

procedure HA(X, Y)
▷ Input vector X of length n indicating the capacity of n classes.
▷ Input vector Y of length m indicating the capacity of m classes.
    If $classCap[i] \leq Y[j]$
        $C(i, j) = Y[j] - classCap[i]$
    Else:
        $C(i, j) = \infty$

B. Baseline with relaxation

Instead of matching the exact class capacity with the classroom capacity. We allow classes to take a slightly larger room to ensure every classes have reasonable spaces for waitlist students. It is almost impossible to know the exact the number of waitlisted students for each class. However, the number of waitlisted students is roughly proportional to the

max enrollment (capacity). Therefore, for simplicity we assume the number of waitlisted students is proportional to the classs max enrollment.

The idea here is to increasing the class capacity by a percentage iteratively and run Hungarian Algorithm (HA) on the newly increased capacity until the HA can no longer solve this problem, as illustrated in Algorithm 1. Therefore at this point, we have a solution that assigns every class to a classroom with reasonable relaxation.

Algorithm 2 HARelax

procedure HARELAX(X, Y)
▷ Input vector X of length n indicating the capacity of n classes.
▷ Input vector Y of length m indicating the capacity of m classes.
    relax = 0.05
    res = None
    while True **do**
        classCap = X * (1+relax):
        If $classCap[i] \leq Y[j]$
            $C(i,j) = Y[j] - classCap[i]$
        Else:
            $C(i,j) = \infty$
        $tempRes = HA(C)$
        If $tempRes == None$:
            Return res
        Else:
            $res = tempRes$
            $relax+ = 0.05$

C. Dynamic Hungarian Algorithm

1) Location Constraint: Apart from the capacity constraints, We would like to assign courses in the same department to same or fewer buildings as explained in the introduction. This is a non-trivial problem since our cost function will have two different type of constraints combined. There are two main difficulties. First how can we quantify and formulate the location constraint into cost function. Second, what is the relative importance of these two constraints; I.E. how should we balance these two constraints in order to achieve a optimal solution.

2) Dynamic Cost matrix for Location Constraint: In this section, we designed a way to quantify the location constraint into a cost matrix. Create a $n \times m$ matrix A, with rows indicating department and columns indicating building. Then $A[i,j]$ means the weight of assigning a class from department i to a classroom in building j. We will normalize the weight into probabilities by dividing the row sum, as shown in equation (1).

We initialize this matrix A by having a larger weight on the originally assigned building for this department (for example, math department will have a larger initialize weight in Wean Hall) as shown in equation (2). Thus we want the classes in each department to have more probability to be assigned in their original assigned building at the beginning of the our algorithm. We update this matrix A every time after we run the HA based on the current assignment situation. I.E. we will increase A[i,j] if a class in department i has been assigned to a classroom in building j.

$$Prob(i,j) = \frac{A[i,j]}{\sum_{j=1}^{m} A[i,j]} \tag{1}$$

$$A[i,j] = \begin{cases} 0.5 & : \textit{if i and j originally matched} \\ 0.1 & : \textit{otherwise} \end{cases} \tag{2}$$

3) Dynamic Hungarian Algorithm: In order to solve the second problem introduced in our previous subsection, we introduced a tradeoff variable and optimized this variable the same way as we solved the relaxation variable by running the HA iteratively.

In previous section, we introduced that we run HA for each time slot selected. However for each time slot we will run HA on a different cost matrix since we keep updating the cost matrix and tradeoff variable while running the HA. Thus we call this model Dynamic Hungarian Algorithm (DHA, see Algorithm 2). Note in the pseudo code we do not include the relaxation part for simplicity but in the actual implementation we included the relaxation.

Algorithm 3 DHA

procedure DHA(X, Y)
▷ Input vector X of length n indicating the capacity of n classes.
▷ Input vector Y of length m indicating the capacity of m classes.
    initialize matrix A_{pq}
    tradeoff = 5
    res = None
for each time slot selected:
    If $classCap[i] \leq Y[j]$
      $C(i,j) = Y[j] - classCap[i] + tradeoff * A[p_i, q_j]$
    Else:
      $C(i,j) = \infty$
    $Res = HA(C)$
    If res has too much wasted space:
      tradeoff = tradeoff * 0.7
      rerun HA(C)
    Else if res has some departments assigning to too many buildings:
      tradeoff = tradeoff * 1.2
      rerun HA(C)
    Else:
      update A using Res
End For loop

V. Result

TABLE 1 shows a portion of the result that our code provided. And this provide the class title the course room and the new room that we assign the class to. From the course number (the classes of the same college always have the same two numbers at the beginning), we can tell that the courses from the same college are in the same building. We can tell that we have improved CMUs classroom assignment by comparing our assignment to the original assignment.

For example for Biomedical Engineering Laboratory(3206) and Biochemistry I are now both in Wean Hall. But according to CMUs assignment, Biomedical Engineering Laboratory was in EDS 125 and Biochemistry was in POS A35 which were not in the same building. Another example is for Introduction to Chemical Engineering and Unit Operations of Chemical Engineering. They are now both in DH according to the new classroom assignment that we made, but Introduction to Chemical Engineering used to be in DH 2302, and Unit Operations of Chemical Engineering used to be in SH 125 which are pretty far away from each other.

TABLE I
Portion of Final Classroom Assignment

Classroom	Course	Course Name
WEH 6327	3206	Biomedical Engineering
WEH 7316	3232	Biochemistry I
WEH 3701	3346	Experimental Neuroscience
DH 2302	42101	Intro to Biomedical Engineering
DH 1209	42203	Biomedical Engineering
DH 1117	42444	Medical Devices
DH A200	42681	Complex Disease Models
DH 1302	42744	Medical Devices
TEP 2612	70105	Business Leadership Endeavor: Intern
TEP 2611	70257	Optimization for Business
TEP 2613	70350	Acting for Business
TEP 2610	70374	business analytics
TEP 2701	70495	Corporate Finance
DH A100	6100	Chemical Engineering
DH 1212	6361	Unit Operations of Chemical Engineering
DH 2105	6704	Advanced Heat and Mass Transfer
WEH 5403	9208	Organic Synthesis and Analysis
WEH 4625	9221	Introduction to Chemical Analysis
WEH 5336	9507	Nanoparticles
WEH 6423	9707	Nanoparticles
DH A303	12100	Civil and Environmental Engineering
DH 2303	12216	Civil and Environmental Engineering
DH 4201	12714	Environmental Life Cycle Assessment

VI. Potential Error and Future Improvement

If we were given more time to work on the project, we could add more features into our algorithm. For example, we might retrieve suggested schedules for all majors on CMU website since we are not authorized to access each student's real schedule. We could try to evaluate all the suggested schedules and minimize the walking distance by considering distance between buildings and forming a Shortest Path problem.

Besides, we could also work more about how to determine the class size to accommodate waitlist students in different ways. For example, for computer science and machine leaning courses, usually there would be more students on the waitlist than courses in other departments. We could retrieve historical data on waitlist history for each course to determine the percentage we need to add for courses in each department.

VII. Conclusion

Our problem was to find an optimal classroom assignment for courses in Spring 2019. We designed our algorithm based on leaving reasonably space for different classes and minimizing the walking distance for students.

We first tried the baseline Hungarian algorithm, which only considers minimizing the empty seats for each class. We then improved our model by increasing the class size to allow each class will have reasonable spaces for waitlist students. We increased the class capacity by a percentage and run Hungarian Algorithm on the newly increased capacity.

To incorporate walking distance to our mode, we designed a way to quantify the location constraint to our model. We initialize the cost matrix and want it have a larger weight on the originally assigned building for this department by introducing a tradeoff variable. We then kept updating the cost matrix and tradeoff variable to get the optimized result.

Our solution with the methods listed above were not guaranteed to be an assignment with the shortest walking distance for each student. We didn't account for students' individual course schedule. These methods found a way to make courses in the same department having relatively shorter walking distance by assigning them in only one or very few buildings. However, the methods could be applied to universities other than Carnegie Mellon.

ACKNOWLEDGMENT

Under supervision of Dr. Alan Frieze, math department, Carnegie Mellon University.

References

[1] H. W. Kuhn, The Hungarian method for the assignment problem, in NRL, vol. 2, page 83-97, 1955.

appendix

Appendix

The complete final assignment generated by our algorithm are presented below.

Room	Class
WEH 6327	3206
WEH 7316	3232
WEH 3701	3346
DH 2302	42101
DH 1209	42203
DH 1117	42444
DH A200	42681
DH 1302	42744
TEP 2612	70105
TEP 2611	70257
TEP 2613	70350
TEP 2610	70374
TEP 2701	70495
DH A100	6100
DH 1212	6361
DH 2105	6704
WEH 5403	9208
WEH 4625	9221
WEH 5336	9507
WEH 6423	9707
DH A303	12100
DH 2303	12216
DH 4201	12714
DH MA341	12757
HBH 1002	67250
HBH 2003	67364
PH 100	73240
PH 107E	73359
HH 1305	18310
HH B103	18491
HH A104	18540
HH C105	18578
WEH 5312	18701
POS 152	18702
HH A101	18723
HH B131	18732
HH A301	18733
POS 153	18745
TEP 2700	18817
TEP 3801	18845
DH 4303	19411
DH 3200	19534
DH 3207	19711
TEP 2702	19714
TEP 3808	19734
PH A19	76101
BH 140C	76101
BH 145C	76106
BH 255B	76106
BH 336A	76107
BH 342F	76107
BH 136A	76108
BH A51	76108
PH 223D	76270
PH 225B	76280
BH 154A	76360
BH 150	76365
PH 226A	76378
PH 226B	76439
BH A54	76465
PH 226C	76760
PH A17	76778
PH A18A	76839
BH 140CE	79209

Room	Class
BH 154A	76360
BH 150	76365
PH 226A	76378
PH 226B	76439
BH A54	76465
PH 226C	76760
PH A17	76778
PH A18A	76839
BH 140CE	79209
BH 140CF	79261
BH 140E	79270
PH A18B	79344
BH 140F	79359
PH A18C	79431
BH 235A	84390
BH 136E	84690
DH C302	27205
DH A302	27217
DH A324	27591
WEH 5201	27752
WEH 5310	27791
POS 151	21112
WEH 7500	21260
POS 160	21270
WEH 5302	21400
WEH 5202	21420
WEH 5207	21604
WEH 5304	21640
DH 2210	24101
DH 2122	24104
WEH 5320	24200
WEH 5328	24203
DH A331	24231
WEH 5415	24282
DH 2315	24311
DH 1211	24441
DH A301D	24671
WEH 5421	24757
DH A325	24778
DH 3302	24783
BH 336B	82115
BH 340A	82115
BH A60N	82162
PH A19A	82245
PH A19C	82271
BH 235B	82273
BH 237B	82278
BH A53	82283
PH A19D	82440
BH 255A	80245
PH A20A	80419
BH 229A	80536
PH A20	80719
PH A21	80836
WEH 5409	33104
WEH 4623	33228
WEH 5316	33342
WEH 4709	33767
PH A21A	85446
PH A22	85706
PH A7C	85746
MI 301B	2510
MI 304	2710
GHC 4401	15110

Room	Class
GHC CLSTR	15112
GHC 4211	15112
GHC 4301	15150
GHC 5207	15150
GHC 4101	15150
GHC 4307	15259
GHC 5222	15365
GHC 4102	15386
MM 103	15418
GHC 4215	15618
WEH 7201	15686
GHC 4303	15750
SH 208	5392
SH 212	5436
SH 222	5499
SH 214	5540
SH 125	5640
SH 219	5692
MM C4	5836
SH 220	5840
POS 146	5872
POS 147	5899
WEH 7218	11422
WEH 7423	11722
MI Auditorium	10707
NSH 3002	16350
POS A35	16385
NSH 3206	16778
NSH 1305	16833
PH 7F	88302
PH 125B	88406
PH 107H	88415
PH 125C	36635
PH 125D	36636
PH 126A	36735
PH 27	36736
SH 208	3350
SH 219	42612
WEH 8427	70100
BH 136E	70122
BH 140C	70321
BH 140CE	70381
BH 140CF	70421
BH 140E	70462
BH 140F	70485
BH 145C	6261
BH 150	6679
BH 154A	6720
BH 136A	6802
BH 229A	9204
BH 235A	12631
BH 235B	67373
WEH 8201	73230
SH 220	73274
SH 222	18290
TEP 2610	18320
BH 237B	18500
BH 255A	18500
BH 255B	18500
BH 336A	18500
HBH 1204	18615
TEP 2611	18639
BH 336B	19213
BH 340A	19356

Room	Class
BH 342F	19425
BH A51	19625
BH A53	19639
BH A54	19756
BH A60N	76100
DH 1117	76101
DH 1209	76101
DH 1211	76106
DH 1212	76106
DH 1302	76107
DH 2105	76107
DH 2122	76108
DH 2210	76108
DH 2302	76245
DH 2303	76265
DH 2315	76270
DH 3200	76271
DH 3207	76318
DH 3302	76330
DH 4201	76363
DH 4303	76718
DH A100	76730
DH A200	76824
DH A301D	79201
DH A302	79213
DH A303	79341
DH A324	79387
DH A325	27520
DH A331	27720
DH C302	21101
DH MA341	21122
GHC 4101	21238
GHC 4102	21241
GHC 4211	21260
GHC 4215	21272
GHC 4301	21341
GHC 4303	21373
GHC 4307	21484
GHC 4401	21721
GHC 5207	24203
GHC 5222	24203
GHC CLSTR	24441
HBH 1002	24730
HBH 2003	24773
HH 1305	82112
HH A101	82122
HH A104	82172
HH B103	82173
HH B131	82201
HH C105	82212
HH A301	82231
MI 301B	82262
MI 304	82265
MI Auditorium	82286
MM 103	82323
MM C4	82332
NSH 1305	82342
NSH 3002	82343
NSH 3206	80100
PH 100	80311
PH 107E	80611
PH 107H	33122
PH 125B	33152
PH 125C	33232

Room	Class
PH 125D	33339
PH 126A	85219
PH 223D	85355
PH 225B	85422
PH 226A	85484
PH 226B	2250
PH 226C	15112
PH 27	15112
PH 7F	15150
PH A17	15150
PH A18A	15150
PH A18B	15210
PH A18C	15390
PH A19	15410
PH A19A	15462
PH A19C	15605
PH A19D	15662
PH A20	15712
PH A20A	15780
SH 214	5392
PH A21	5571
PH A21A	5671
PH A22	5692
PH A7C	11634
POS 146	11634
POS 147	10701
POS 151	16311
POS 152	16455
POS 153	16681
POS 160	16831
POS A35	36200
SH 125	36490
SH 212	36602
SH 208	3412
SH 219	3755
BH 150	42202
BH 136E	42613
BH 255B	42630
BH 140F	42678
BH 229A	42679
BH 140CE	70100
BH 140CF	70110
BH 154A	70207
BH 140E	70257
BH 235A	70412
BH 336B	70443
BH 340A	70495
BH 235B	12271
BH A53	12358
BH 140C	67272
BH 237B	73102
BH 255A	73103
BH A54	73315
BH 336A	18202
SH 214	18441
BH 342F	18447
BH A51	18482
DH 2303	18625
DH 2315	18632
DH 3302	18690
DH A100	18731
DH A301D	18741
DH A303	18751
DH 3200	18795

Room	Class
HH 1305	19402
DH 3207	19605
HH A101	19722
GHC 5207	76100
GHC 5222	76101
PH A22	76101
SH 222	76106
TEP 2610	76106
PH 107H	76107
PH 125B	76107
PH 125C	76107
PH 125D	76107
PH 226B	76108
PH 226C	76108
PH 27	76108
PH A19C	76108
DH 4201	76241
DH 4303	76265
DH A200	76270
DH A302	76358
DH A324	76394
MI 304	76460
NSH 1305	76758
PH 126A	76798
POS A35	79104
PH 223D	79218
PH 225B	79302
TEP 2612	79307
PH 226A	84389
PH 7F	84689
TEP 2613	27101
BH A60N	27570
DH 1117	27725
TEP 2701	21111
TEP 2702	21120
TEP 3808	21124
WEH 4623	21259
WEH 4625	21261
WEH 4709	21301
WEH 5310	21660
WEH 5328	21832
HH A104	24200
HH A301	24352
DH 1209	24653
DH 1212	24692
DH 1302	24786
PH A17	82101
PH A18A	82102
PH A18C	82122
PH A19A	82131
PH A19D	82132
PH A20	82135
PH A20A	82232
PH A21	82235
PH A21A	82278
PH A7C	82345
POS 147	82372
SH 125	82416
WEH 5336	80150
WEH 5403	80180
SH 212	80256
WEH 5421	33446
WEH 6327	33756
WEH 6423	85102

Room	Class
WEH 7201	2201
WEH 7218	15112
WEH 7423	15112
SH 220	15150
TEP 2611	15150
DH C302	15150
PH A18B	15312
DH MA341	15464
GHC 4101	15664
GHC 4102	15745
GHC 4215	5418
GHC 4301	5818
GHC 4303	5899
GHC 4307	11796
HBH 1002	11797
HBH 2003	10301
MI 301B	10315
MI Auditorium	10601
MM 103	10708
MM C4	16791
NSH 3002	16824
PH 100	88221
PH 107E	88417
POS 146	36207
POS 151	36315
POS 153	36700
BH 136A	3124
BH 140C	3750
BH 136E	42647
BH 140CE	70110
BH 145C	70122
BH 140CF	70381
BH 140E	70415
BH 140F	70422
BH 150	70471
BH 154A	70493
BH 255B	6100
BH 235A	6364
BH 336A	12232
BH 342F	12358
BH 235B	12600
BH 237B	12657
BH 255A	12768
BH 336B	67202
BH 340A	67250
BH A53	67315
BH A54	73341
BH A51	18748
BH A60N	19303
DH 2303	19703
DH A100	19704
DH 1117	19733
DH 1209	19867
DH A302	76101
DH 1212	76270
DH 1302	76310
DH 3200	76347
DH 3207	76420
DH 4201	76475
DH 4303	76747
DH A200	76820
GHC 4101	76875
GHC 4102	79203
GHC 4215	79339

Room	Class
GHC 4301	84402
GHC 4303	84602
DH A324	27202
DH A331	27212
DH C302	27542
DH MA341	27740
GHC 4211	21127
GHC 4401	21256
GHC CLSTR	21703
PH A19	21801
POS 151	21820
POS 160	24231
GHC 4307	24659
GHC 5207	80330
HBH 1002	80382
HBH 1204	80618
HBH 2003	80630
MI 301B	80682
WEH 3701	33448
MI 304	85102
WEH 5202	85102
MI Auditorium	85310
MM 103	85320
MM C4	85763
WEH 5207	15110
WEH 5302	15112
WEH 5304	15112
NSH 1305	15150
NSH 3002	15150
PH 100	15150
PH 107E	15381
WEH 5310	15494
WEH 5316	15694
PH 107H	15721
PH 125B	5320
PH 125C	5410
PH 125D	5410
PH 126A	5410
PH 223D	5410
PH 225B	5610
PH 226A	5610
PH 226B	5610
PH 226C	5610
PH 27	5820
PH 7F	11423
PH A17	11623
PH A18A	11727
PH A18C	11775
PH A19A	11823
PH A19C	10405
PH A19D	10605
PH A20	16264
PH A20A	16865
PH A21A	88367
PH A22	88380
WEH 5415	36726
BH 136E	3231
BH 140C	3232
BH 140CE	42341
BH 140CF	42624
BH 140E	70205
BH 140F	70205
BH 145C	70207
BH 150	70350

Room	Class
BH 229A	70455
BH 235A	70480
BH 235B	6665
BH 237B	6702
BH 255A	9737
BH 255B	12704
BH 336A	18883
BH 336B	18883
BH 340A	19351
BH 342F	76101
BH A51	76106
BH A53	76106
BH A54	76107
BH A60N	76107
DH 1117	76108
DH 1209	76108
DH 1211	76270
DH 1212	76361
DH 1302	76761
DH 2105	79265
DH 2122	79415
DH 2210	27217
DH 2302	27741
DH 2303	21122
DH 2315	21127
DH 3200	21268
DH 3207	21269
DH 3302	21344
DH 4201	21355
DH 4303	21630
DH A100	21702
DH A200	24334
DH A301D	24687
DH A302	24704
DH A303	82111
DH A324	82141
DH C302	82142
DH MA341	82162
GHC 4101	82171
GHC 4102	82172
GHC 4211	82202
GHC 4215	82215
GHC 4301	82241
GHC 4303	82242
GHC 4307	82272
GHC 4401	82304
HH 1305	33120
HH A101	33121
HH A104	33142
HH B103	33211
HH B131	33213
HH C105	33234
HH A301	33466
MI 301B	33765
MI 304	85102
MI Auditorium	85211
MM 103	2613
MM C4	15112
NSH 1305	15112
NSH 3002	15150
NSH 3206	15150
PH 100	15150
PH 107E	15292
PH 107H	15296
PH 125B	15351

Room	Class
PH 125C	15650
PH 125D	5317
PH 126A	5470
PH 223D	5617
PH 225B	5670
PH 226A	11688
PH 226B	11785
PH 226C	10301
PH 27	10601
PH 7F	16785
PH A17	36202
PH A18A	36207
PH A18B	36226
PH A18C	36226
PH A19	36326
BH A51	3124
DH MA341	70365
PH A18B	9101
BH 136E	9101
BH 140C	9208
BH 140CE	9563
BH 140CF	9760
BH 140E	9763
BH 140F	12745
BH 145C	67306
BH 150	67445
BH 154A	18100
BH 229A	18220
BH 235A	18240
BH 235B	18600
BH 237B	76404
BH 255A	76471
BH 255B	76494
BH 336A	76794
BH 336B	76804
BH 340A	76899
BH 342F	79277
BH A53	79319
BH A54	84372
BH A60N	84672
DH 1117	27722
DH 1209	24200
DH 1211	24200
DH 1212	24281
DH 1302	24281
DH 2105	24341
DH 2122	82103
DH 2210	82104
DH 2302	82133
BH 136A	82888
DH 2303	33104
DH 2315	85102
DH 3200	15251
DH 3207	15294
DH 3302	15295
DH 4201	15394
BH 145C	6363
DH 1211	9106
DH 2105	9220
DH 2122	9345
DH 2210	73102
DH 2302	76101
DH A325	76101
DH A331	76106
GHC 4211	76106

Room	Class
GHC 4401	76107
GHC CLSTR	76107
HBH 1204	76108
HH B103	76108
HH B131	79305
HH C105	21228
NSH 3206	21236
PH 107H	21240
PH 125B	21241
PH 125C	21242
PH 125D	21356
PH 226B	21369
PH 226C	21374
PH 27	21623
PH A19	82121
PH A19C	82174
POS 152	82332
POS A35	82362
TEP 2612	80100
TEP 2613	80135
TEP 2700	33114
TEP 2701	33332
TEP 2702	15112
TEP 3801	15112
TEP 3808	15312
WEH 3701	36757
BH 136A	3121
PH A18B	3121
POS 146	3344
POS 151	9105
POS 153	9221
DH 1211	12351
DH 2105	73103
BH 145C	18663
DH 2122	76101
DH 2210	76101
DH 2302	76106
DH A325	76106
DH A331	76107
GHC 4211	76107
GHC 4401	76108
GHC CLSTR	76108
HBH 1204	76239
HH B103	79104
HH B131	27100
HH C105	21127
NSH 3206	21256
PH A19	21292
POS 152	21325
POS 160	21329
TEP 2700	21355
TEP 3801	21723
WEH 3701	21737
WEH 5201	24231
WEH 5202	33104
WEH 5207	33332
WEH 5302	33771
WEH 5304	15112
WEH 5312	15112
WEH 5316	15312
BH 136E	3713
BH 140C	6663
BH 140CE	9714
BH 140CF	9912
BH 140E	12603

Room	Class
BH 140F	18600
BH 145C	18600
BH 150	18709
BH 154A	18757
BH 229A	18759
BH 235A	19681
DH 2122	76101
BH 235B	21997
BH 237B	24104
BH 255A	24104
DH 2210	24231
BH 255B	24618
BH 336A	24672
BH 336B	24691
BH 340A	82131
BH A53	82132
BH A54	82172
BH A60N	82272
DH 2302	15112
DH 2315	15112
DH 1117	15150
DH 1209	15150
DH 1211	15150
DH 1212	15719
DH 1302	5681
DH 2105	16730
BH 136A	9218
BH 136E	12231
BH 140C	76101
BH 145C	21241
BH 255A	21259
BH 342F	82162
BH A51	82292
BH A53	33141
BH A54	33444
DH 2303	67272
DH 2315	33104
DH 2122	85406
DH 2210	85806
BH 342F	15112
BH A51	15112

```
department=c(79,85,12,27,73,88,3,33,2,11,42,18,24,76,82,36,9,15,10,6,19,16,67,80,21,5,70,84)

building = c("BH", "CFA", "CIC", "CYH", "DH", "EDS", "GHC", "HBH", "HH", "HL", "MI", "MM",
"NSH", "PH", "POS", "SH", "TEP", "WEH")

cost_mat <- matrix(0, nrow = length(department), ncol = length(building))
colnames(cost_mat) = building
rownames(cost_mat) = department

weight_1 = 5
weight_2 = 3

cost_mat[1,"BH"] = weight_1
cost_mat[1,"PH"] = weight_2
cost_mat[2,"BH"] = weight_1

cost_mat[2,"PH"] = weight_2
cost_mat[3,"DH"] = weight_1
cost_mat[4,"DH"] = weight_1

cost_mat[5,"PH"] = weight_1
cost_mat[5,"BH"] = weight_2
cost_mat[6,"PH"] = weight_1

cost_mat[6,"BH"] = weight_2
cost_mat[7,"WEH"] = weight_1
cost_mat[8,"WEH"] = weight_1

cost_mat[9,"GHC"] = weight_1
cost_mat[9,"MI"] = weight_2
cost_mat[10,"GHC"] = weight_1

cost_mat[11,"DH"] = weight_1
cost_mat[12,"HH"] = weight_1
cost_mat[13,"DH"] = weight_1

cost_mat[14,"BH"] = weight_1
cost_mat[14,"PH"] = weight_2
cost_mat[15,"BH"] = weight_1

cost_mat[15,"PH"] = weight_2
cost_mat[16,"PH"] = weight_1
cost_mat[16,"BH"] = weight_2
```

```
cost_mat[17,"WEH"] = weight_1
cost_mat[18,"GHC"] = weight_1
cost_mat[19,"GHC"] = weight_1
cost_mat[20,"DH"] = weight_1
cost_mat[21,"DH"] = weight_1

cost_mat[22,"NSH"] = weight_1
cost_mat[23,"HBH"] = weight_1
cost_mat[24,"BH"] = weight_1

cost_mat[24,"PH"] = weight_2
cost_mat[25,"WEH"] = weight_1
cost_mat[26,"SH"] = weight_1

cost_mat[26,"GHC"] = weight_2
cost_mat[27,"TEP"] = weight_1
cost_mat[27,"POS"] = weight_2

cost_mat[28,"BH"] = weight_1
cost_mat[28,"DH"] = weight_2


update_cost <- function(count, df){
 for(i in 1:nrow(df)){
  depart = as.character(as.integer(new_assign$course[i]/1000))
  building = substr(new_assign$new_room[i],1,3)
  if(substr(building,3,3) == " "){
   building = substr(building,1,2)
  }
  count[depart,building] = count[depart,building] + 1
 }
 return(count)
}


# Find the most popular 30-min time slot
all_schedule$Days = as.character(all_schedule$Days)

extract = function(data, lookup){
 str = as.character(data[i,lookup])
 t_1 = as.numeric(substr(str, 1,2))
 t_2 = as.numeric(substr(str, 4,5))
 t_apm = substr(str, 6,7)
 if(t_apm == "AM" || t_1 == 12){
```

```
   slot = (t_1-8)*2 + t_2/30
  }
  else {
   slot =  8 + (t_1)*2 + t_2/30
  }
  return(slot)
}

time_slot = rep(NA, nrow(all_schedule))
for(i in 1:nrow(all_schedule)){
  slot_s = extract(data = all_schedule, lookup = "Begin")
  slot_e = extract(data = all_schedule, lookup = "End")
  time_slot[i] =  list(slot_s:slot_e)
}

all_schedule$Slots = time_slot
wed = subset(all_schedule, regexpr("W", all_schedule$Days) != -1)

sort(table(unlist(wed$Slots)), decreasing = TRUE)[1]
# 13 -> 2:30 PM is the most popular time slot on Wed


sub_set = function(df, slot){
  res = c()
  for(i in 1:nrow(df)){
    if(slot %in% df$Slots[[i]]){
      res = c(res, i)
    }
  }
  return(res)
}

# Subset of courses taking place at 2:30 PM on Wed
wed_13_courses = wed[sub_set(wed,13),]



library("clue")
library("assertthat")

penalty = 5
max_enroll=all_schedule$Capacity
room=all_schedule$Room
```

```
classroom_cap = na.omit(classroom$Capacity)
classroom_names = classroom[1:178,1]

wed_13_max_enroll = wed_13_courses[,"Capacity"]
wed_13_classrooms = wed_13_courses[,"Room"]
cost=matrix(0,nrow = length(wed_13_max_enroll),ncol=length(classroom_cap))

cost_row = rowSums(cost_mat)
for(i in 1:length(wed_13_max_enroll)){
  for (j in 1:length(classroom_cap))
    if (classroom_cap[j]-wed_13_max_enroll[i]>=0){
      depart = as.character(as.integer(wed_13_courses$Course[i]/1000))
      building = substr(classroom$Room[j],1,3)
      if(substr(building,3,3) == " "){
        building = substr(building,1,2)
      }

      building_cost = 1 - cost_mat[depart, building]/as.numeric(cost_row[depart])
      cost[i,j]= penalty*building_cost
       + classroom_cap[j]-wed_13_max_enroll[i]
    }
    else{
      cost[i,j]=9999
    }
}

assign_room = solve_LSAP(cost, maximum = FALSE)
new_assign = data.frame(cbind(seq_along(assign_room), assign_room))
new_assign$new_room = classroom[new_assign$assign_room,1]
new_assign$course = wed_13_courses$Course
new_assign$title = wed_13_courses$Title
new_assign$slots = wed_13_courses$Slots



cost_mat = update_cost(cost_mat, new_assign)
wed_not_courses = wed
wed_popular_time = 13

while (nrow(new_assign) < nrow(wed)){
  wed_not_courses = wed_not_courses[-sub_set(wed_not_courses,wed_popular_time),]

  wed_not_max_enroll = wed_not_courses[,"Capacity"]
```

```
  wed_popular_times = sort(table(unlist(wed_not_courses$Slots)), decreasing = TRUE)

  wed_popular_time = as.integer(names(wed_popular_times)[1])

  wed_popular_time_courses =
wed_not_courses[sub_set(wed_not_courses,wed_popular_time),]

  cost = matrix(0,nrow=nrow(wed_popular_time_courses),ncol=length(classroom_cap))

  cost_row = rowSums(cost_mat)
  for(i in 1:nrow(wed_popular_time_courses)){
    course_i_slot = wed_popular_time_courses[i, "Slots"]
    for (j in 1:length(classroom_cap)){
      if (as.character(classroom_names[j]) %in% as.character(new_assign$new_room)){
        overlap_rooms = grep(paste("^",as.character(classroom[j,1]),"$", sep=""),
as.character(new_assign$new_room))
        assert_that(length(overlap_rooms) >= 1)
        for (k in 1:length(overlap_rooms)){
          overlap_room_index = overlap_rooms[k]
          if (length(intersect(course_i_slot[[1]], new_assign[overlap_room_index, "slots"][[1]])) != 0){
            cost[i,j] = 9999
          }
        }
      }
      else if (classroom_cap[j]-wed_not_max_enroll[i]>=0){
        depart = as.character(as.integer(wed_13_courses$Course[i]/1000))
      building = substr(classroom$Room[j],1,3)
      if(substr(building,3,3) == " "){
        building = substr(building,1,2)
      }

      building_cost = 1 - cost_mat[depart, building]/as.numeric(cost_row[depart])
      cost[i,j]= penalty*building_cost
       + classroom_cap[j]-wed_13_max_enroll[i]
    }
    else{
      cost[i,j]=9999
    }
    }
  }
  assign_room = solve_LSAP(cost, maximum = FALSE)
  temp_new_assign = data.frame(cbind(seq_along(assign_room), assign_room))
  temp_new_assign$new_room = classroom[temp_new_assign$assign_room,1]
  temp_new_assign$slots = wed_popular_time_courses$Slots
```

```
  temp_new_assign$course = wed_popular_time_courses$Course
  temp_new_assign$title = wed_popular_time_courses$Title
  cost_mat = update_cost(cost_mat, temp_new_assign)
  new_assign = rbind(new_assign, temp_new_assign)
}

cost_mat
cost_row
```