Optimization of Course Scheduling

Grace Ekstrom, Brandon Tang, Isabelle Telford, Brendan Willis

December 13, 2022

Abstract

The University of Massachusetts mathematics department creates a new course schedule every semester. The process of manually creating a schedule by hand is long and tedious due to the large amount of data and factors that need to be considered when creating the course schedule. In this paper, we propose a linear optimization code that will construct a new course schedule for the mathematics department that will maximize each professor's schedule preferences. Then, the program will take last-minute changes that need to be made to that schedule, and minimize further changes that are made to the original schedule while ensuring that the necessary changes are made.

1 Introduction

Our motivation for this project is to create a more efficient way of constructing a course schedule that will maximize instructor preferences while satisfying constraints born from availability, qualification, room allocation restrictions, course section dispersion, and other restrictions. Along with our schedule builder, our goal is to create a logical way of adapting to last minute changes that will affect the original schedule as little as possible.

The University of Massachusetts mathematics department is made up of four different types of instructors who range in the number of courses they teach each semester. There are lecturers that usually teach three courses in the fall and spring, VAPs that usually teach two course in the fall and spring, tenure-stream professors that usually teach two courses in one semester and one in the other, and then graduate students, some of whom teach one course during a semester. Permanent lecturer, VAP, and professor assignments are created well ahead of the semester for which the schedule is being made, but some sections of low-level or large lecture courses are left open to be assigned for newly hired faculty and graduate students closer to the beginning of the semester. Before scheduling begins, the instructors fill out a survey with their schedule preferences. This survey includes what type of instructor they are and their preferences for days, times, and courses. In our project, we weighted each instructor's preferences to allow us to find an optimal solution for the schedule that maximizes these preferences. Then, once we had constructed an optimal schedule, we chose a set of changes that are representative of schedule change requests the department receives, and adjusted the optimal schedule to adhere to these changes. The goal is to create as few other changes to the schedule as possible while satisfying the new changes, as well as the constraints put on the schedule building, as we do not want the new schedule to be unrealistic.

2 Literature Review

To prepare for the creation of our model, we read several papers on linear programs to build schedules for university departments. In papers on scheduling from the University of Philippines [2], referred to as paper 1 from here on, and California State Polytechnic University Pomona [1], referred to as paper 2, both groups aim to create a schedule that maximizes preferences for faculty members. In paper 1, the preferences are based on the course, while in paper 2, the preferences are based on the time slot. Both papers ensure that each course is taught by one faculty member, but paper 1 treats instructors and professors explicitly separately, while paper 2 relies on hard-coding in differences between levels of teachers. For our project, we follow the model of paper 1, as we have lecturers, VAPs, and tenure-stream professors and each category has significant differences in the types and amounts of courses they can teach. In addition, the variables used in the papers are different. In paper 1, for a given type of teacher, there is one binary variable for each possible pair of teacher and class. If it is 1, that person is teaching that class, and if it is 0, they are not. In paper 2, there is one binary variable for each possible combination of person, class, and time-slot that is 1 if that person is teaching that class at that time-slot, and 0 if not. This difference in variable leads to a slight difference in the setup of the objective function, as in paper 1, they sum over the product of preference and variable for each person and class. In paper 2, they first get the sum of the variables for a certain person over all courses to see if that person is teaching in that time slot, which is a new binary value of 1 if the person is teaching at that time and 0 if not, then multiply that new value by the preference and sum over all time slots and people. Because of the different types of weights in the papers, it makes sense that the two want to consider different binary variables. We have preference data for both course and time, so we want to create an objective function for our initial schedule creation that maximizes both aspects, while using assignment variables as in paper 2.

Paper 2 assumes that courses already have set time-slots, which could be helpful for our project as post-registration, we want to avoid changing when a certain course is taught. Paper 1 does not have this constraint, which is more helpful in the first part of our project, when we want to build a schedule from scratch.

In paper 1, the authors assign minimum and maximum loads for instructors and professors, and ensure that each person stays within that range. They do that by looking at how many units or credits a certain course is, how many sections of that course a person is teaching, and then adding that up over all courses for each person. In paper 2, the authors require each person to teach exactly a given number of classes. While this number can be different for every person, having this type of constraint would make it very difficult to make changes to a given schedule, but as our data includes a set number of courses each instructor should teach, our model follows a mix of both approaches.

Paper 2 has an explicit constraint to make sure a person only teaches 2 or 3 days a week, which we would like to enforce for our project for most people. We decide to allow this to be broken for lecturers, but ideally this would hold for all people. Both papers also have constraints to make sure that no one teaches both early in the morning and late in the evening on any given day. In fact, paper 2 designed their objective function to create compact schedules, although they had to do that by hard-coding in preferences for compact days. We would also like to make sure that no one teaches both early and late, so we mimic these constraints in our program.

Paper 2 handles core courses (lower-level courses required for many students) in a useful way. The constraints for the number of sections of each core course taught allows for not all sections needing to be assigned to a faculty member, and the remaining sections are later assigned to part-time lecturers. In our case, we want to mimic this so new lecturers and graduate students have classes to teach.

Paper 2 also has constraints to make sure that different sections of core courses are spread out, which allows for more choices for students' schedules so everyone who needs a class is able to enroll in a section. We want to consider this in our project as well, as there are quite a few lower-level courses for which we want variation in scheduling. In addition, we want to add constraints that keep courses that students often take during the same semester from being scheduled at the same time.

As noted, there are many similarities in the constraints that we use in our project and the constraints that were used in papers 1 and 2, however there are some aspects of our project that were not handled by either paper. At the core of this is the change minimizer, which, despite having similar constraints to the schedule builder, has a different objective function and several additional constraints that do not apply to the papers' goals.

3 Model

3.1 Schedule Builder

3.1.1 Data and Parameter Collection

The heart of this project is the schedule builder, as not only does it produce the output that the change minimizer works on, its constraints are the basis for the constraints of the change minimizer. For this project, it is important to make a distinction between "course" and "class". A "course" is, for example, MATH 100; it is the umbrella term for any section teaching the same content. A course itself does not have a time and instructor assignment; those are made at the "class," or "section" level. Each course has a set number of sections, and each of these classes needs to be assigned a time and instructor. This will be especially important in understanding how the constraints integrate the data, as we have information at both levels that need to be handled differently. To understand the problem space, we needed to know what kind of data the department has for each faculty member, so we understood what kind of decisions it makes when assigning people to classes and time slots. We want our linear program to mimic these decision making processes as best as possible, and as we were not in direct conversation with the department, the data and an initial briefing were our main sources of the constraint concepts we would need. The department sends a survey to each instructor to get preferences on different aspects of scheduling, and one crucial part of our project was to figure out how to format the data produced by this survey. The data we were given was in the form of a spreadsheet that contained, for each instructor:

1. their level (VAP, lecturer, or professor),
2. how many classes an instructor can teach in a given semester,
3. upper-level classes they would want to teach most, second most, and third most,
4. a ranking for each large lecture class from 1-3,
5. whether or not they volunteer to teach large lectures ,
6. lower-level classes they would want to teach most, second most, and third most,
7. classes they would not like to teach,
8. whether they would prefer to teach multiple different courses or multiple sections of the same course,
9. whether they would like to teach courses back-to-back,
10. ranking of times they'd like to teach (morning, midday, or afternoon) from 1-3, with chances to exclude extremely early or late courses if early or late respectively was ranked first,
11. ranking of days (MWF, MW, TuTh, MTWTh, MTWThF) they would want to teach from 1-5

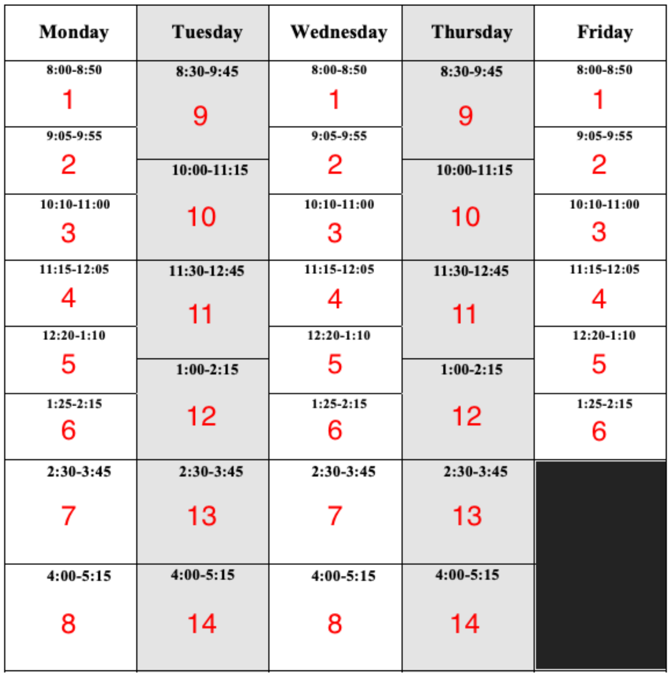
12. whether they have specific, valid reasons for only being able to teach two days a week if they ranked that first, and

13. any times they were unable to teach.

With this information we generated the data sets we would use for our program.

The first data set was for course preferences. This data set contained 70 rows corresponding to each instructor and 61 columns corresponding to each course. A class got a value of 1 if the instructor ranked it as their top choice for that type of course, 0.75 if they ranked it second, and 0.5 if they ranked it third. If the instructor volunteered to teach large lectures then the values of their rankings were the same as above, but if they didn't volunteer, their first ranked large lectures were given a value of 0.3, their second ranked given 0.2, and third ranked given 0.1. Any classes that the lecture said they didn't want to teach was given a 0, and any classes not mentioned at all were given 0.25. We note that these decisions are inherently a bit arbitrary, but as we do not have enough information to know how an instructor views each course, we were forced to make a decision that, at the very least, showed relative preferences that matched what information we did have. Ideas for how to improve this process will be discussed in the conclusion.

The second data set was for time preferences. Using the department's scheduling blocks, we made 14 time slots.



Monday	Tuesday	Wednesday	Thursday	Friday
8:00-8:50 1	8:30-9:45 9	8:00-8:50 1	8:30-9:45 9	8:00-8:50 1
9:05-9:55 2	10:00-11:15 10	9:05-9:55 2	10:00-11:15 10	9:05-9:55 2
10:10-11:00 3		10:10-11:00 3		10:10-11:00 3
11:15-12:05 4	11:30-12:45 11	11:15-12:05 4	11:30-12:45 11	11:15-12:05 4
12:20-1:10 5	1:00-2:15 12	12:20-1:10 5	1:00-2:15 12	12:20-1:10 5
1:25-2:15 6		1:25-2:15 6		1:25-2:15 6
2:30-3:45 7	2:30-3:45 13	2:30-3:45 7	2:30-3:45 13	
4:00-5:15 8	4:00-5:15 14	4:00-5:15 8	4:00-5:15 14	

Figure 1: Time Slots for the UMass Math Department

Early time slots are 1-3 and 9-10, midday time slots are 4-6 and 11-12, and late time slots are 7-8 and 13-14.

To determine the preference for each particular time slot, we considered their relative ranking of MWF, MW, and TTh (MTWTh and MTWThF were handled separately), and their ranking of early, midday, and late. Using 1 as the highest ranking, we multiplied their ranking for each possible day and time combination, as shown below.

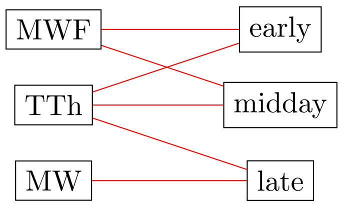


Figure 2: Possible Day-Time Combinations

Once the products were calculated, each time slot for every instructor was given a weight corresponding to the product. The possible weights and their associated products were 1 (1), .8 (2), .6 (3), .4 (4), .2 (6), or 0 (9). Again, this makes many assumptions about the preferences of professors, including the fact that each person views days and times as equally important. Alternate ways of approaching this are discussed in the conclusion.

In addition, we received information about each course including the type (large lecture, lower-level, upper-level, graduate), any professor that was already assigned to teach it, and groups of classes that are often taken during the same semester, so we can ensure these courses do not all overlap. While the preference data is used in the model's objective function, this data as well as some of the other data from the instructor spreadsheet is referenced in constraints the model attempts to satisfy, as will be shown below.

3.1.2 Objective Function

The goal of our schedule builder program was to create a schedule that maximizes professors' course and time preferences. To create our objective function, we first needed variables that would assign each professor to a class and time slot. These assignment variables will be binary, meaning they will be equal to 1 if person i is teaching class j (at the class/section level, not course) at time h, and 0 if they are not.
Assignment Variables:

- x_{ijh} = VAP $i \in \{1, \dots, n_1\}$ teaching class $j \in \{1, \dots, m\}$ at time $h \in \{1, \dots, l\}$
- y_{ijh} = lecturer $i \in \{1, \dots, n_2\}$ teaching class j at time h
- z_{ijh} = tenure-track professor $i \in \{1, \dots, n_3\}$ teaching class j at time h

The objective function also needs to incorporate each professors preference weight for both class and time.
Preference Weight Parameters:

- a_{ij} = preference weight for person i teaching class j
- b_{ih} = preference weight for person i teaching at time h.

To handle the fact that course preferences are at the course level, but we need to consider assignments at the class level, we force a_{ij} to have the value of the preference that instructor had for the course that j is a section of.

In a linear program with so many potentially conflicting aspects, we need to allow some of our constraints to be broken if necessary. To make this mathematical, we use softness or slack variables in constraints that can be broken to adjust the value that some function of our variables is related to (equal to, less than or equal to, etc.). In this case, our softness variables can take on binary and non-negative integer values, depending on the kind of constraint we are working with and how much we want to allow it to change.

After deciding our variables, we initially created an objective function that summed over all assignments, our x, y, z variables, each multiplied by the corresponding a and b parameter, then subtracted off all softness variables. In essence, as our assignment variables are binary, this produces an optimal value that represents the sum of all $a_{ij}b_{ih}$ that correspond to an ijh combination that was assigned, minus the amount by which any of our constraints were not satisfied. It is clear that it is optimal to have softness variables take on values as close to 0 as possible, but why a and b are multiplied is less intuitive. To multiply them was inherently an arbitrary choice, and upon realizing this, we decided to test whether it was a good choice by also formulating the objective function with some other function of the preferences that would be maximal when a and b themselves are maximal. We chose $a + b$, partially because this opens the door to the ability to give more weight to time or course preferences if in the future we find out that people care more about one or the other. We note that the two versions of the objective function create different "ideal" situations. Remembering that $0 \leq a, b \leq 1$, consider the situations where, for a certain instructor, time, and class:

1. $a = 0.6, b = 0.2$
2. $a = 0.3, b = 0.4$
3. $a = 0.7, b = 0.1$

Situations 1 and 2 have the same product, so our first objective function would treat them the same, but our second objective function would prefer situation 1 over situation 2 as their sums are not the same. On the other hand, situations 1 and 3 have the same sum, but their products are different and our first objective function would prefer situation 1 to situation 2 while the second objective function would see them as equally optimal. In general, multiplying the weight gives preference to a and b values that are closer together, while adding them gives preference to values that are more extreme, For example, adding the preferences would output a time that is highly preferred but a course that is not over a time and course that are both somewhat preferred.

There are plenty of other ways to combine the time and weight preferences in the objective function, but choosing a more complicated function of the two was harder to justify. That said, a project that attempted to find an optimal function of the two that produces the output with the highest overall preferences could be interesting for future students.

Schedule 1 Objective Function:

$$\max \quad \sum_{i=1}^{n_1}\sum_{j=1}^{m}\sum_{h=1}^{l} a_{ij}b_{ih}x_{ijh} + \sum_{i=1}^{n_2}\sum_{j=1}^{m}\sum_{h=1}^{l} a_{ij}b_{ih}y_{ijh} + \sum_{i=1}^{n_3}\sum_{j=1}^{m}\sum_{h=1}^{l} a_{ij}b_{ih}z_{ijh} - \text{softness variables} \tag{1}$$

Schedule 2 Objective Function:

$$\max \quad \sum_{i=1}^{n_1}\sum_{j=1}^{m}\sum_{h=1}^{l} (a_{ij}+b_{ih})x_{ijh} + \sum_{i=1}^{n_2}\sum_{j=1}^{m}\sum_{h=1}^{l} (a_{ij}+b_{ih})y_{ijh} + \sum_{i=1}^{n_3}\sum_{j=1}^{m}\sum_{h=1}^{l} (a_{ij}+b_{ih})z_{ijh} - \text{softness variables} \tag{2}$$

Where n_1 = the number of VAPs, n_2 = the number of lecturers, n_3 = the number of tenure-track professors, m = the number of classes, and l = the number of time slots.

3.1.3 Constraints

1. *Each course should be assigned exactly one time slot and one instructor, with the exception of some lower-level courses or large lectures which may be filled in by new instructors later.* A common theme in our constraints is summing over certain indices, sometimes multiple, and looking at the assignments with that index or combination of indices. As the assignment variables are binary and equal 1 when that combination of class, time, and instructor is assigned, how we set up that sum allows us to find out something about individual assignments or assignments belonging to a certain category (time, instructor, class, course, etc.). In this constraint, we want to pick a class, then add up all the assignments that class has. Unless we don't assign it, that sum needs to be 1, and for upper-level and graduate courses, the sum must be exactly 1 as we need to assign it. As we do not want our program to leave all lower-level and large lecture sections unassigned, we introduce our first soft variable, p_1, but force it to take on 0 (i.e. not be soft) in the situation that a class is upper-level or graduate.

$$\sum_{i=1}^{n_1}\sum_{h=1}^{l} x_{ijh} + \sum_{i=1}^{n_2}\sum_{h=1}^{l} y_{ijh} + \sum_{i=1}^{n_3}\sum_{h=1}^{l} z_{ijh} = 1 - p_{1j} \qquad \forall j \tag{3}$$

$$p_{1j} = 0 \qquad \forall j \in \{\text{upper-level or grad class}\} \tag{4}$$

$$p_{1j} \in \{0,1\} \qquad \forall j \tag{5}$$

2. *Each person teaches at most one class during any time-slot.* We need to be able to pick any person and any time, and sum over all the assignments with that combination of person and time and make sure it is at most 1, so they are either teaching at that time (= 1), or not teaching at that time (= 0). The sum will never be negative as our variables cannot be negative, so we represent this by saying the sum is at most 1.

$$\sum_{j=1}^{m} x_{ijh} \le 1 \qquad \forall i \in \{1,\dots,n_1\}, h \tag{6}$$

$$\sum_{j=1}^{m} y_{ijh} \le 1 \qquad \forall i \in \{1,\dots,n_2\}, h \tag{7}$$

$$\sum_{j=1}^{m} z_{ijh} \le 1 \qquad \forall i \in \{1,\dots,n_3\}, h \tag{8}$$

3. *Pre-assigned course-instructor c, i pairings are upheld.* Certain courses have already been assigned an instructor before the rest of scheduling is done, and it is important that our output respects that. This assignment is at the course level, but sometimes that instructor is supposed to teach multiple sections of a course, so we want to take all assignments for that person and sections of that course at any time, and have that sum equal the number of sections of that course that they are supposed to teach.
Let $C = \{j : j \text{ is a section of course } c\}$

$$\sum_{j\in C}\sum_{h=1}^{l} x_{ijh} = r_{ic} \tag{9}$$

if the instructor is a VAP,

$$\sum_{j\in C}\sum_{h=1}^{l} y_{ijh} = r_{ic} \tag{10}$$

if the instructor is a lecturer, and

$$\sum_{j\in C}\sum_{h=1}^{l} z_{ijh} = r_{ic} \tag{11}$$

if the instructor is a tenure-track professor, where r_{ic} = the number of sections of course c instructor i is teaching.

4. *Instructors that cannot teach certain courses should not be assigned to teach those courses.* For various reasons, an instructor may not be qualified or able to teach a certain course. We want the total assignments of that person and the sections of that course at any time to be 0, meaning they are not teaching it at all.
   For fixed i^* and c^* being the instructor-course pair that cannot be assigned, and $C^* = \{j : j \text{ is a section of course } c^*\}$:

$$\sum_{h=1}^{l}\sum_{j\in C^*} x_{i^*jh} = 0 \tag{12}$$

if the instructor is a VAP,

$$\sum_{h=1}^{l}\sum_{j\in C^*} y_{i^*jh} = 0 \tag{13}$$

if the instructor is a lecturer, and

$$\sum_{h=1}^{l}\sum_{j\in C^*} z_{i^*jh} = 0 \tag{14}$$

if the instructor is a tenure-track professor.

5. *Instructors that cannot teach at certain times should not be assigned to teach during those times.* Similarly, some instructors have previous commitments or other aspects of their job that happen at certain times, and it is important that they are not scheduled to teach at that time. So for that person and time, any assignment for any class should be 0, meaning the sum should also be 0.
   For fixed i^* and h^* being the instructor-time pair that cannot be assigned:

$$\sum_{j=1}^{m} x_{i^*jh^*} = 0 \tag{15}$$

if the instructor is a VAP,

$$\sum_{j=1}^{m} y_{i^*jh^*} = 0 \tag{16}$$

if the instructor is a lecturer, and

$$\sum_{j=1}^{m} z_{i^*jh^*} = 0 \tag{17}$$

if the instructor is a tenure-track professor.

6. *Lecturers cannot teach upper-level or graduate classes.* Lecturers are typically hired to teach lower-level and lecture classes, and because of that often times are not qualified to teach the upper-level and graduate classes. This can actually be ignored in some specific situations, but our model does not consider this, and a future version may want to allow for that. For this version, if you take any lecturer and any upper-level or graduate class, the assignments at any time, and thus the sum, should be 0.
$$\sum_{h=1}^{l} x_{ijh} = 0 \qquad \forall i \in \{1, \dots, n_1\}, j \in \{\text{upper-level or grad}\} \tag{18}$$

7. *Each instructor teaches the correct number of courses.* As this number cannot be generalized for everyone, we will assign each person a parameter L_{xi}, L_{yi}, or L_{zi} accordingly that represents the load or number of classes that person is able to teach. In addition, it may be necessary to not fully satisfy this constraint at times, so we introduce another soft variable p_2. We made two assumptions here: that an instructor would not want to teach more classes than they are supposed to, and that they would not want to reduce their load by more than one. Based on these assumptions, we keep the p_2s binary, and subtract them from the associated load parameter, so the total number of classes a person teaches is their load or one less. We want to pick a person and add all their assignments for any class and time combination, and have that equal their load minus the softness variable.
$$\sum_{j=1}^{m} \sum_{h=1}^{l} x_{ijh} = L_{xi} - p_{2xi} \qquad \forall i \in \{0, \dots, n_1\} \tag{19}$$
$$\sum_{j=1}^{m} \sum_{h=1}^{l} y_{ijh} = L_{yi} - p_{2yi} \qquad \forall i \in \{0, \dots, n_2\} \tag{20}$$
$$\sum_{j=1}^{m} \sum_{h=1}^{l} z_{ijh} = L_{zi} - p_{2zi} \qquad \forall i \in \{0, \dots, n_3\} \tag{21}$$
$$p_{2xi}, p_{2yi}, p_{2zi} \in \{0, 1\} \quad \forall i \tag{22}$$

8. *Instructors should not be assigned to teach both MWF/MW and TTh unless they want to.* In our creation of time preference weights, we handled each instructor's ranking of MWF, MW, and TTh classes, as we know it is uncommon for instructors to want to teach both sets of days. In almost every case, we want to make it so that no instructor is scheduled to teach 4 or 5 days a week. Mathematically we can enforce that by taking an instructor and one time slot on MW(F) and one time slot on TTh, and setting the sum of all class assignments with that person and either time to at most 1. If we do this for each pair of MW(F) classes and TTh classes, we force the program to only assign an instructor to classes on one set of days. In the case that a person does want to teach 4 or 5 days a week (because of all the responsibility of tenure-track professors, we only ever see this with VAPs and lecturers, but the constraint could easily be adapted to allow full-week teaching for professors as well), we assigned a parameter q based on the survey data that takes a value of 1 if the professor ranked 4 or 5 days a week as their top choice, and 0 otherwise. When we add this to 1 one the right hand side, this allows a person to be assigned to teach during both a MW(F) and TTh time slot.
$$\sum_{j=1}^{m} x_{ijh'} + \sum_{j=1}^{m} x_{ijh''} \leq 1 + q_{xi} \qquad \forall i, h', h'' \tag{23}$$

$$\sum_{j=1}^{m} y_{ijh'} + \sum_{j=1}^{m} y_{ijh''} \leq 1 + q_{yi} \qquad \forall i, h', h'' \tag{24}$$

$$\sum_{j=1}^{m} z_{ijh'} + \sum_{j=1}^{m} z_{ijh''} \leq 1 \qquad \forall i, h', h'' \tag{25}$$

Where $h' = \{1, \ldots, 8\}$, $h'' = \{9, \ldots, 14\}$, and $q_{xi}, q_{yi} = 1$ if a VAP or lecturer respectively would like to teach all 5 days, 0 otherwise.

9. *Instructors should not be assigned to teach both early MWF and late MW.* The first and last class of a given day are quite far apart, and we do not want anyone to have to teach at both times. For MW(F), for any professor, this means not having an assignment for any class at both the first time slot and the last time slot on MW, so we take the sum of assignments for that person and the first time slot, and the sum of assignments for that person and the last MW time slot, and add those together. This final sum should be at most one, meaning the person is teaching during neither time slot or one, but not both. For lecturers, we make this constraint soft as it is common for lecturers to teach 3 classes, and this can make it difficult to give them short days. For softness, we make a variable p_3 for each lecturer that is binary. We add it to the right-hand side to say that the person can be assigned to teach during both the first and last time slot of MW(F)s.

$$\sum_{j=1}^{m} x_{ij1} + \sum_{j=1}^{m} x_{ijw} \leq 1 \qquad \forall i \tag{26}$$

$$\sum_{j=1}^{m} y_{ij1} + \sum_{j=1}^{m} y_{ijw} \leq 1 + p_{3i} \qquad \forall i \tag{27}$$

$$\sum_{j=1}^{m} z_{ij1} + \sum_{j=1}^{m} z_{ijw} \leq 1 \qquad \forall i \tag{28}$$

$$p_{3i} \in \{0, 1\} \quad \forall i \tag{29}$$

Where $w =$ the last MW time slot.

10. *Instructors should not be assigned to teach both early and late TTh.* We need the same type of constraint for TTh, with the same assumptions and reasoning. We have another binary soft variable p_4 for each lecturer that acts the same was p_3 does above.

$$\sum_{j=1}^{m} x_{ij(w+1)} + \sum_{j=1}^{m} x_{ijl} \leq 1 \qquad \forall i \tag{30}$$

$$\sum_{j=1}^{m} y_{ij(w+1)} + \sum_{j=1}^{m} y_{ijl} \leq 1 + p_{4i} \qquad \forall i \tag{31}$$

$$\sum_{j=1}^{m} z_{ij(w+1)} + \sum_{j=1}^{m} z_{ijl} \leq 1 \qquad \forall i \tag{32}$$

$$p_{4i} \in \{0, 1\} \quad \forall i \tag{33}$$

Where $w =$ the last MW time slot, $l =$ the last TTh time slot.

11. *For each time slot, there should be an appropriate number of lower- and upper-level courses being taught so we can assign classes to rooms.* There are several reasons to want to disperse classes throughout the possible time slots, and they have different mathematical implementations. One concrete reason is because of room assignments, a process that happens after time/instructor scheduling. At UMass, upper-level and graduate classes are held in Lederle Graduate Research Center and Tower, while lower-level and lecture classes are often held in other buildings. For this reason, we want to have one set of constraints for each half of the classes. Additionally, we want to handle "popular" time slots differently from the earliest and latest time slots that are unpopular among both students and teachers. We handle everything that is not the first or last time slot of a day first. Letting β be the number of these time slots, we want to find the average number of classes for each of these time slots. If α is the number of lower-level and lecture classes, the average of these classes is $\frac{\alpha}{\beta}$, and the average of the rest of the classes (the upper-level and graduate classes), is $\frac{m-\alpha}{\beta}$. As this average may not be an integer, we want to keep all the assignments for each time slot between the floor and ceiling of the average. If it is an integer, this in effect makes the constraint an equality. This constraint is allowed to be soft to allow more or fewer classes to be taught if needed, but the program was taking advantage of this, so we penalize breaking this constraint by multiplying the value of the softness variable p_4, which can take on integer values between 0 and 6, by 1000 in the objective function so slight changes make large differences to the objective value.

$$\left\lfloor \frac{\alpha}{\beta} \right\rfloor - p_{41h} \leq \sum_{i=1}^{n_1}\sum_{j=1}^{\alpha} x_{ijh} + \sum_{i=1}^{n_2}\sum_{j=1}^{\alpha} y_{ijh} + \sum_{i=1}^{n_3}\sum_{j=1}^{\alpha} z_{ijh} \leq \left\lceil \frac{\alpha}{\beta} \right\rceil + p_{42h} \quad \forall h \in \{\text{not first or last of the day}\} \tag{34}$$

$$\left\lfloor \frac{m-\alpha}{\beta} \right\rfloor - p_{43h} \leq \sum_{i=1}^{n_1}\sum_{j=\alpha+1}^{m} x_{ijh} + \sum_{i=1}^{n_2}\sum_{j=\alpha+1}^{m} y_{ijh} + \sum_{i=1}^{n_3}\sum_{j=\alpha+1}^{m} z_{ijh} \leq \left\lceil \frac{m-\alpha}{\beta} \right\rceil + p_{44h} \quad \forall h \in \{\text{not first or last of the day}\} \tag{35}$$

$$p_{41h}, p_{42h}, p_{43h}, p_{44h} \in \{0, 1, \ldots, 6\} \quad \forall h \tag{36}$$

Where α = the number of lower-level and large lecture sections being taught, and β = the number of time slots that are not the first or last of the day.

12. *For less-popular time slots, we don't need to lower-bound the number of classes.* This constraint is very similar to the previous one, but we do not need to have a minimum number of classes for the first and last time slots of each day. We again allow this to be soft with non-negative integer $p_5 \leq 6$, but it is also heavily penalized.

$$\sum_{i=1}^{n_1}\sum_{j=1}^{\alpha} x_{ijh} + \sum_{i=1}^{n_2}\sum_{j=1}^{\alpha} y_{ijh} + \sum_{i=1}^{n_3}\sum_{j=1}^{\alpha} z_{ijh} \leq \left\lceil \frac{\alpha}{\beta} \right\rceil + p_{51h} \quad \forall h \in \{\text{first or last of the day}\} \tag{37}$$

$$\sum_{i=1}^{n_1}\sum_{j=\alpha+1}^{m} x_{ijh} + \sum_{i=1}^{n_2}\sum_{j=\alpha+1}^{m} y_{ijh} + \sum_{i=1}^{n_3}\sum_{j=\alpha+1}^{m} z_{ijh} \leq \left\lceil \frac{m-\alpha}{\beta} \right\rceil + p_{52h} \quad \forall h \in \{\text{first or last of the day}\} \tag{38}$$

$$p_{51h}, p_{52h} \in \{0, 1, \ldots, 6 \quad \forall h \tag{39}$$

13. *We do not want sections of core courses (lower-level, large lecture) to overlap too much.* The number of sections of a given course that can overlap will depend on how many sections of that class there are, so we assign a parameter:
s_c= max number of sections of core course c that can be taught during a time-slot.
For each time slot and core course, we want all assignments for each class belonging to that class, and we want to make sure that the sum is at most that s_c. Let $C = \{j : j \text{ is a section of course } c\}$

$$\sum_{j\in C}\sum_{i=1}^{n_1} x_{ijh} + \sum_{j\in C}\sum_{i=1}^{n_2} y_{ijh} + \sum_{j\in C}\sum_{i=1}^{n_3} z_{ijh} \le s_j \quad \forall h, c \in \{\text{core courses}\} \tag{40}$$

14. *We want courses that are commonly taken together to have little overlap so that students are able to fit all of those courses in their schedule.* For some sets of courses that each have many sections, there can be some overlap between one class and another, but for upper-level courses that have only one or two sections, we want these sections to be at different times. To do this, we want to take each set of courses and each time slot, and then, for each section of a course in that set, we want all the assignments for that class and time slot. Once we have that for each section, we add them all together. If the set of courses contains courses with lots of sections, we want this sum to be at most 2, meaning two classes in that set are taught at that time slot. If the courses have fewer sections, however, we only want at most one class being taught at each time slot.
Take sets $D_1, D_2, \dots$ to be sets of j that belong to lower-level courses often taken together.

$$\sum_{j\in D_k}\left(\sum_{i=1}^{n_1} x_{ijh} + \sum_{i=1}^{n_2} y_{ijh} + \sum_{i=1}^{n_3} z_{ijh}\right) \le 2 \qquad \forall h, k \tag{41}$$

Take sets $F_1, F_2, \dots$ to be sets of j that belong to upper level courses often taken together.

$$\sum_{j\in F_k}\left(\sum_{i=1}^{n_1} x_{ijh} + \sum_{i=1}^{n_2} y_{ijh} + \sum_{i=1}^{n_3} z_{ijh}\right) \le 1 \qquad \forall h, k \tag{42}$$

15. *If someone is up for promotion but has not taught a variety of courses yet, we want them to teach at least one new course during this semester.* Universities only want to promote someone that adds lots of value to the faculty, part of which is being able to teach a variety of courses. People that are able to be up for promotion are VAPs and lecturers, and we want to take each person who is up for promotion and also needs to teach a new course this semester to get ensure they have enough experience. We want to find the sum of all the classes they are teaching that are sections of courses they have previously taught, and make sure that number is less than the number of classes they are teaching. This would imply that they are teaching some other course than the ones they have previously taught. As our program is implemented in the form of code using Gurobi, we cannot have strict inequalities, so we account for this by adding 1 to the number of classes they are teaching that they have already taught, and make the inequality less than or equal to their total assignments.
Let
$\alpha_i = \{c : \text{person } i \text{ has taught course } c \text{ before}\}$
$C_{\alpha_i} = \{j : j \text{ is a section of a course in } \alpha_i\}$

$$\sum_{j\in C_{\alpha_i}}\sum_{h=1}^{l} x_{ijh} + 1 \le \sum_{j=1}^{m}\sum_{h=1}^{l} x_{ijh} \tag{43}$$

$$\forall i \in \{\text{VAPs to be promoted and needs new courses}\}$$

$$\sum_{j \in C_{\alpha_i}} \sum_{h=1}^{l} y_{ijh} + 1 \leq \sum_{j=1}^{m} \sum_{h=1}^{l} y_{ijh} \tag{44}$$

$$\forall i \in \{\text{lecturers to be promoted and needs new courses}\}$$

16. *To ensure that courses for which some sections are left open still have a qualified chair, we want at least one of the sections to be taught by a tenure-track professor or permanent lecturer.* Lower-level and large lecture courses often have multiple sections and are coordinated courses, meaning that one person teaching a section acts as course chair and is the leader of all instructors of that course. This person needs to be qualified, and UMass gives that distinctions to professors and lecturers. This constraint is allowed to be broken if necessary, so we introduce p_6, a binary softness variable. For each lower-level or large lecture course, we take all the professor and lecturer assignments for sections of that course and set it to at least 1, minus the p_6 for that course. If there was a VAP who was pre-assigned to teach one of these classes, we assume they are qualified to be chair, so we set p_6 for that course equal to 1, meaning the constraint is forced to not have to hold.

$$\sum_{i=1}^{n_3} \sum_{j \in C} \sum_{h=1}^{l} z_{ijh} + \sum_{i=1}^{n_2} \sum_{j \in C} \sum_{h=1}^{l} y_{ijh} \geq 1 - p_{6c} \tag{45}$$

$\forall c \in \{\text{lower-level or big lecture}\}$

$$p_{6c} \in \{0, 1\} \quad \forall c \tag{46}$$

3.2 Change Minimizer

3.2.1 Data and Parameter Collection

The second portion of our project is the change minimizer. This is another linear program based largely on the schedule builder in both the sense that it has many of the same constraints, and in the sense that it relies on the output of the schedule builder. At this stage, we assume a schedule has been created, and that the faculty members have received their assignments and have had the chance to tell the department about any issues with their schedules. Oftentimes, issues arise with the time someone is scheduled to teach; they may have had a new commitment come up that has a set time that they were unaware of before filling out the original survey. Due to the automated nature of our schedule builder, it is possible that someone is assigned to teach a class they do not feel they are qualified for, and needs to be taken off that assignment. It is also possible that someone needs to suddenly go on leave, and can no longer teach at all. While there are other possible changes that could theoretically arise, our program is designed to handle those mentioned above. The goal of this program is to accommodate those last-minute changes and create a new schedule that has as few differences as possible to the original schedule.

There is no time frame that specifies a "last-minute change"; this means the department could be notified about the changes either before student registration begins or after students have already enrolled in classes. Knowing this, we added one constraint for each registration status to allow us to prioritize certain changes, which will be explained below.

3.2.2 Objective Function

The same x, y, z assignment variables and similar softness variables are used in the change minimizer as in the schedule builder, along with new change variables. The change variables are binary variables for each instructor-class-time combination that will equal 1 if there was a change from the original schedule and 0 if it stayed the same.
Change Variables:

- $\hat{x}_{ijh}$ = whether the status of VAP i teaching class j at time h has changed
- $\hat{y}_{ijh}$ = whether the status of lecturer i teaching class j at time h has changed
- $\hat{z}_{ijh}$ = whether the status of tenure-track professor i teaching class j at time h has changed

In order to keep our new schedule as similar to the original schedule as possible, we want to minimize the number of assignments that have changed. That means we want the total of our change variables to be as small as possible, so we find the sum of all of them. We want to penalize breaking any soft constraints, so we also add our softness variables. As our objective is to minimize, adding these non-negative values will be considered non-optimal to the program. We chose only to add the softness variables that come from new constraints for two reasons. First, we assume the goal of change minimizing will help to somewhat artificially enforce the schedule builder constraints and minimize their softness. Our outputted schedule from the builder had to be optimal, and we are trying to not move away from that schedule. Second, we may be more willing to break some of the schedule builder constraints to a higher degree now that we have changes to take into account, so by not penalizing them, we tell the program to take changes to the schedule more seriously than softness of schedule builder constraints.

Change Minimzer Objective Function:

$$\min \quad \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{h=1}^{l}\hat{x}_{ijh} + \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{h=1}^{l}\hat{y}_{ijh} + \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{h=1}^{l}\hat{z}_{ijh} + \text{ new softness variables} \tag{47}$$

3.2.3 Constraints

1. *The $\hat{x}, \hat{y}, \hat{z}$ take on the correct values.* We will take an already-created schedule, and assign the following binary (1 = yes, 0 = no) parameters from that:
   $\overline{x}_{ijh}$ = lecturer i teaches class j at time h in original schedule
   $\overline{y}_{ijh}$ = VAP i teaches class j at time h in original schedule
   $\overline{z}_{ijh}$ = tenure-track professor i teaches class j at time h in original schedule
   Then we want the $\hat{x}, \hat{y}, \hat{z}$ variables to take on 0 if the new and original schedule have the same assignment, and 1 if they are different, so we have a linear manipulation:

$$\hat{x}_{ijh} = x_{ijh} + \overline{x}_{ijh} - 2\overline{x}_{ijh}x_{ijh} \qquad \forall i, j, h \tag{48}$$

$$\hat{y}_{ijh} = y_{ijh} + \overline{y}_{ijh} - 2\overline{y}_{ijh}y_{ijh} \qquad \forall i, j, h \tag{49}$$

$$\hat{z}_{ijh} = z_{ijh} + \overline{z}_{ijh} - 2\overline{z}_{ijh}x_{ijh} \qquad \forall i, j, h \tag{50}$$

2. *All constraints from the schedule builder will apply for this program as well.*

3. *For instructor i^* who has an issue with teaching course c^*, we add a constraint as in constraint 4 of the schedule builder.* These constraints say that that person does not teach that course. In the coded model, we do this by simply adding to the list of instructor-course pairs that cannot be assigned. Mathematically, we just need to add a new equality constraint.

4. *For instructor i^* who has an issue with teaching at time h^*, we add a constraint as in constraint 5 of the schedule builder.* These constraints say that that person does not teach at that time. Again, we do this by adding to a list of pairs that cannot be assigned in the code, but by adding another constraint explicitly in the mathematical representation.

5. *An instructor suddenly on leave does not teach at all.* For an instructor i^* who is suddenly on leave, we need all their assignments, and thus their sum, to add to 0, so we add a constraint

$$\sum_{j=1}^{m}\sum_{h=1}^{l} x_{i^*jh} = 0, \tag{51}$$

$$\sum_{j=1}^{m}\sum_{h=1}^{l} y_{i^*jh} = 0, \text{ or} \tag{52}$$

$$\sum_{j=1}^{m}\sum_{h=1}^{l} z_{i^*jh} = 0 \tag{53}$$

to match the level of instructor they are.

6. *If pre-registration, all courses c should keep their original instructor i^* and have the time changed.* If students have not yet begun to register for courses when we get the list of changes to be made, we would rather instructors keep the courses they were originally assigned to teach, and have the time change instead. This constraint is soft, and is not implemented if the instructor-course pair was one that was asked to be changed. For a given instructor, if they had been assigned to teach a course in the original schedule, we want them to have at least 1 assignment for some section of that course at some time. We cannot guarantee that this is going to be possible, so this is soft with variables p_7, p_8, p_9 for each course and instructor; these are binary, and if they get assigned a value of 1 it lets the left hand side already be at least 1 without an assignment being made.
Where $C = \{j : j \text{ is a section of course type } c\}$,

$$\sum_{j\in C}\sum_{h=1}^{l} x_{i^*jh} + p_{7ic} \geq 1 \qquad \forall c \in \{\text{courses } i^* \text{ was teaching in orig. schedule}\} \tag{54}$$

$$\sum_{j\in C}\sum_{h=1}^{l} y_{i^*jh} + p_{8ic} \geq 1 \qquad \forall c \in \{\text{courses } i^* \text{ was teaching in orig. schedule}\} \tag{55}$$

$$\sum_{j\in C}\sum_{h=1}^{l} z_{i^*jh} + p_{9ic} \geq 1 \qquad \forall c \in \{\text{courses } i^* \text{ was teaching in orig. schedule}\} \tag{56}$$

$$p_{7ic}, p_{8ic}, p_{9ic} \in \{0, 1\} \quad \forall i, c \tag{57}$$

7. *If post-registration, a course c should keep its time-slot h^* and have the instructor changed.* Once students have begun registering for classes, we need to shift priorities onto them. If they have registered for a certain class at a certain time, they have likely made decisions based on that time slot, so we do not want to move it around. This means that, if we have to make a change, we want it to be to the professor. This is again soft as we will almost definitely need to change at least a few time assignments, so we introduce binary p_{10} for each course and time. This time, we want to take a time slot and a course that was supposed to be taught during it, and find all the assignments for any section of that course by any person at that time. We want to make sure at least one class is being taught then, but it is soft so occasionally the left hand side can be made to be at least one just by the softness variable.

$$\sum_{j\in C}\sum_{i=1}^{n_1} x_{ijh^*} + \sum_{j\in C}\sum_{i=1}^{n_2} y_{ijh^*} + \sum_{j\in C}\sum_{i=1}^{n_3} z_{ijh^*} + p_{10hc} \geq 1 \qquad \forall c \tag{58}$$

$$p_{10hc} \in \{0,1\} \quad \forall h, c \tag{59}$$

4 Computational Results and Analysis

4.1 Small Example

We first tested our program on a small example. For the schedule builder, there are four instructors who need to be assigned to a course–a large lecture (1), a lower level (2), and two upper level (3 and 4)–and a time slot–MWF morning (1), MWF afternoon (2), TTH morning (3), and TTH afternoon (4). Each instructor teaches one course and each course must be taught at a different time.

Data:

- Instructor 1 is a Lecturer, who cannot teach upper level courses.
- Instructor 2 is a VAP.
- Instructor 3 is a Professor who has been already been assigned to teach course 4.
- Instructor 4 is a Professor who can only teach Tuesdays and Thursdays afternoon.

Time Preferences:

Instructor	1	2	3	4
1	1	0.75	0.5	0.25
2	0.75	1	0.25	0.5
3	0.75	0.25	1	0.5
4	0.25	0.5	0.75	1

Course Preferences:

Instructor	1	2	3	4
1	1	0.75	0.25	0.25
2	0.75	1	0.5	0.25
3	0.25	0.5	0.75	1
4	0.25	0.5	1	0.75

We tested this on our version of the schedule builder where time and course preferences were multiplied. Optimal Solution:

$$\begin{aligned}
&x_{111} = 1\\
&x_{222} = 1\\
&x_{343} = 1\\
&x_{434} = 1\\
&(1 \cdot 1)x_{111} + (1 \cdot 1)x_{222} + (1 \cdot 1)x_{343} + (1 \cdot 1)x_{434} = 4
\end{aligned}$$

Above, is the optimal schedule for the instructors' preferences. After students have already enrolled in courses, Instructor 2 had a last minute change and now cannot teach MWF afternoons (time slot 2). The optimal solution that minimizes changes to the original schedule:

$$\begin{aligned}
x_{122} &= 1\\
x_{211} &= 1\\
x_{343} &= 1\\
x_{434} &= 1
\end{aligned}$$

There were 2 changes made to assignments from the original schedule, and 0 changes made to time slots for classes.

And assuming instructor 1 is qualified to be a chair for course 1, our schedule builder code produces the same output as the non-code approach. The change minimizer, however, gives a different output. With the same assumptions as above as well as assuming we want all sections assigned (the first optimal solution left both low-level classes unassigned), the code has the assignments as:

$$\begin{aligned}
x_{122} &= 1\\
x_{231} &= 1\\
x_{343} &= 1\\
x_{414} &= 1
\end{aligned}$$

This may be for many reasons, including the fact that we have slightly changed our objective function to penalize changes differently than was likely calculated by hand, and the fact that there are so many constraints working on the code that may be too difficult to track when doing the work by hand. The by-hand optimal solution is actually better than the one outputted by the code, as there were 3 changes made, 2 of which involve changing the time slot of a class, so we assume that that solution breaks some constraint the code is implementing or that the changes in the objective function made a bigger difference than expected. It is possible that this is an example of the domino effect of changes that may be avoided when changes are made by hand and certain constraints are actively chosen to be broken.

4.2 Schedule Builder

Figures 3 and 4 show the outputted schedules for the version of the schedule builder with preferences multiplied and added, respectively. Note that in both versions, classes are well spread out across time slots, and that it was even optimal to not have any classes early in the morning on MWF, a time slot that neither students nor instructors tend to prefer. While it is good to see a reasonable looking schedule, we also want to take a closer look at individual instructors and try to gauge how good the schedule builder is at giving them a schedule they would like.

Professor 8 was supposed to teach 2 classes this semester. They did not volunteer to teach large lectures; their preferences for time of day was early (but not 8am MWF), midday, then late; and their preferences for day of the week was TTh, MW, MWF. They did not list any classes they did not want to teach or any times they were unable to. The output from Schedule 1 (ab) was teaching 37 (TTh 8:30-9:45) and 18 (TTh 10-11:15), and from Schedule 2 ($a+b$), teaching 37 (TTh 8:30-9:45) and 38 (TTh 10-11:15). Course 37 was their top-ranked upper-level course, and course 18 was their top-ranked lower-level course. It is clear that they should be quite happy with the first schedule, and mostly happy with the second schedule, despite not having ranked course 38.

Lecturer 11 had a load of 3 courses, and while they volunteered to teach large lecture courses, they did not want to teach upper-level courses. They also preferred early classes, then midday, then late, and for day of the week, ranked their choices as TTh, MWF, then MW. They listed no specific times they could not teach. In schedule 1, their assignments were teaching 14 (MWF 9:05-9:55), 14 (MWF 10:10-11), and 16 (MWF 1:25-2:15). In schedule 2, they were assigned to teach 16 (MWF 10:10-11), 14 (MWF 12:20-1:10), and 14 (MWF 1:25-2:15). Course 14 was their third ranked lower-level course, while they did not rank course 16 at all. In this case, both schedules produce the same course assignments with slightly different times. Schedule 1 has slightly earlier time slots for the first two classes, but as both schedules have a 1:25pm class, these earlier classes mean schedule 1 creates a longer day, which might be less ideal even though the person wanted early classes.

Lastly, VAP 2 was supposed to teach 2 classes. They listed many courses as their top-ranked for both upper- and lower-level courses, but do not volunteer for large lectures. While there are no other courses they would prefer not to teach, they cannot teach Tuesdays from 4 to 5pm. Otherwise, they prefer early to midday to late, and TTh to MW to MWF. This is one of many instances where time of day and day of week preferences are somewhat in conflict. VAP 2 ranks early and midday above late, but MW above MWF. MW classes, however, are only offered late, so it is hard to know exactly when the person would rather teach, especially if we are unable to assign them to TTh time slots as these are highly coveted. In this case, both schedules gave them at least one TTh class, but schedule 1 assigned them to teach 27 (MW 4-5:15) and 27 (TTh 10-11:15), so they would have to teach 4 days a week. Schedule 2, however, has them teaching 2 sections of course 27 again, but this time on TTH 8:30-9:45 and TTh 10-11:15. This is likely to be a great schedule for them, as they teach two early morning classes on TTh. In addition, course 27 is ranked second for upper-level courses.

From looking at a few people in depth as well as glancing at more instructors, it seems as though most people are getting at least one course they ranked in their top three for some type of course. Given that there are a lot of common time preferences between instructors, it is harder to give some people their highly-ranked times, but this would be an issue if doing the schedule by hand as well. One possible way the department may accommodate this that our schedule builder does not is by allowing more classes on Tuesdays and Thursdays, which are by far the most popular days to want to teach. Our program handles these time slots the same as any other time, and tries to keep an equal number of classes in every slot across the week. If we think that keeping this spread of classes is important, however, our schedule builder does a good job of creating a reasonable schedule that produces individual schedules that line up well with people's preferences, especially given how many preferences put different people in conflict.

4.3 Change Minimizer

We created two versions of results for each schedule, one where the changes occur before student registration and one for after student registration.

Schedule 1 needed the following changes:

1. VAP 5 teaching course 26
2. VAP 14 teaching at time 7
3. Lecturer 11 teaching at time 3
4. Professor 14 teaching at time 11
5. Professor 2 on leave

Schedule 2 needed the following changes:

1. VAP 5 teaching course 24
2. VAP 14 teaching at time 7
3. Lecturer 11 teaching at time 3
4. Professor 13 teaching at time 10
5. Professor 2 on leave

Figure 5 and 7 show the corresponding changes to schedule 1 and 2 for pre-registration and Figure 6 and 8 show the corresponding schedule and changes for post registration. When comparing the two change minimizer results, we notice the post-registration schedules have much better results. Our goal for post-registration was to change the instructor over the time slot, and as a result, over ninety percent of classes kept the same time post-registration. In schedule 1 (ab), 33 assignments were kept exactly the same, while 108 classes were kept at the same time post-registration. For schedule 2 ($a+b$), 39 were kept the exact same and 108 again kept the same time slot. Post-registration, both schedules 1 and 2 only had 5 assignments stay the exact same. Schedule 1 had 49 classes keep the same instructor, while schedule 2 had 46. This is a sign that it is much easier to keep class-time pairings stable than than it is for class-professor pairings. Thankfully, fewer requests come in to explicitly change the courses a professor is teaching, so it may be easier to avoid some of these ripple-effect changes. The downside to this is that, in general, having changes come in as early as possible should be a good thing. If we run our program earlier, however, it is likely to make more changes than if we run it after registration. The kinds of changes being made pre-registration are the ones that should be in favor of the instructor, as they ideally get to keep their courses, so seeing that this is not actually capable of doing what we want is disappointing.

In general, we do not see a significant difference between schedules created with the multiplied differences versus the added preferences.

To get a better sense of how people are affected by these forced changes, we revisit two people we had initially looked at for the schedule builder.

Lecturer 11 had an issue with teaching MWF 10:10-11, a time they were scheduled to teach in both schedules. As a reminder, in schedule 1 they were scheduled to teach 14 (MWF 9:05-9:55), 14 (MWF 10:10-11), 16 (MWF 1:25-2:15). In the new schedule 1 pre-registration, they are teaching 14 (MWF

8-8:50), and 14 (12:20-1:10). Post registration, they are teaching 14 (MWF 9:05-9:55), 3 (MWF 11:15-12:05), and 14 (MWF 1:25-2:15). In schedule 2, they were originally assigned to teach 16 (MWF 10:10-11), 14 (MWF 12:20-1:10), and 14 (MWF 1:25-2:15). Pre-registration with changes, they are not assigned to teach 14 (TTh 8:30-9:45), and 14 (TTh 2:30-3:45). Post-registration they have 17 (MWF 8-8:50), 14 (MWF 1:25-2:15), and 14 (TTh 8:30-9:45). Courses 3 and 17 are new additions to their schedule in different versions, neither of which they ranked highly. Those replaced course 16, however, which was not ranked highly either, so these swaps are, to the best of our knowledge, neutral. The program successfully took them out of their unwanted time slot, and in all but the schedule 2 post-registration output, kept them at time slots that are about equal to or better than their original ones. We see the program taking advantage of the softness in the schedule builder that allows an instructor's load to decrease by 1 in both pre-registration schedules, which they may not be happy with if they were preparing for a certain workload.

Professor 8 had no forced changes and was happy with their schedule, however their schedule was affected to accommodate the other professors' changes. As stated in the schedule builder results, in schedule 1 professor 8 was teaching course 37 (TTh 8:30-9:45) and course 18 (TTh 10-11:15). In the pre-registration results, professor 8 is scheduled to only to teach course 27 (MWF 8-8:50), another case of courseload reduction. The professor would most likely not be happy with this new schedule because they ranked the MWF 8am time slot rather low, and course 27 was not one they ranked. The post-registration results have professor 8 teaching course 14 (MWF 10:10-11) and course 27 (TTh 8:30-9:45). This is again far from ideal as it means they have to teach classes they did not rank four days a week. In schedule 2, professor 8 was originally scheduled to teach course 37 (TTh 8:30-9:45) and course 38 (TTh 10-11:15). Our change minimizer results now have professor 8 teaching course 44 (MWF 8-8:50) and course 31 (MWF 10:10-11) for pre-registration changes, and teaching course 27 (TTh 8:30-9:45) and course 26 (TTh 1:00-2:15) for post-registration changes. Again, professor 8 is likely to have have preferred their original schedules, as none of the new classes were highly ranked, and any changes to time slots were not in line with their preferences.

5 Conclusion

Having analyzed the outputted schedules, we believe that the department's scheduling process can be automated. With the constraints that we implemented in our program, we already see many instructors being assigned to classes and times that they would be happy with. This is without having implemented everything the department employee takes into account when doing scheduling by hand, but many of these remaining aspects are also able to be expressed mathematically, and will be added in future iterations of this project. As hundreds of hours are currently spent on scheduling just one semester, using code that runs in approximately one minute would save lots of valuable time. This would free up time to tinker with the output by hand and add the 'human touch' our code may not be able to fully replicate. For this reason, we strongly recommend the department implement the schedule builder aspect of our project.

To improve the objectivity of the program, we recommend that the way preference data is collected from faculty members is changed. We found ourselves making many assumptions—for example that any course an instructor did not rank in their top three was equally as preferable to them, and that no one had stronger preferences towards day-of-week versus time-of-day—which we know cannot be true. We had no way of knowing, however, people's true feelings about these, so making an assumption that put things on different footings would have been unfair as well. As there are only 14 time slots, having instructors give each of these a weight expressing how much they want to teach at that time would not be extreme to ask for, and would take out much of the guesswork. Ideally, we would have instructors do the same for courses. There are 61 courses, and this might seem like a lot to weight, but realistically many instructors have a lot of classes they do not want to teach that could be given a weight of 0. Giving thought to each of the remaining courses may seem like a lot of work, but is not much different from what they have to do currently when picking which courses to rank highly. In addition, if the instructors were informed that these few extra minutes would lead to schedules that suited their preferences better, they may be willing to take the time.

The efficacy of the change minimizer in its current state is not what we had hoped, and while there is surely a way to implement it better, we are not sure that it is ready for use. Post-registration it may have some value, and at least provide a way to visualize some swaps and changes that could be made, but we assume the department would prefer if more than about 30% of assignments stayed the exact same. We would be interested in knowing how last-minute changes are currently handled to see if there is a way to implement those methods as a linear program that is more stable than ours. Pre-registration, there are so few assignments that remain the same that it is difficult to recommend our code is used. It seems to make many changes to satisfy constraints, when it is possible that by hand, a swap would be made that does break a constraint, but for the sake of avoiding the domino effect we are seeing in our output. Changes to the change minimizer as a whole may improve its efficacy pre-registration as well, however.

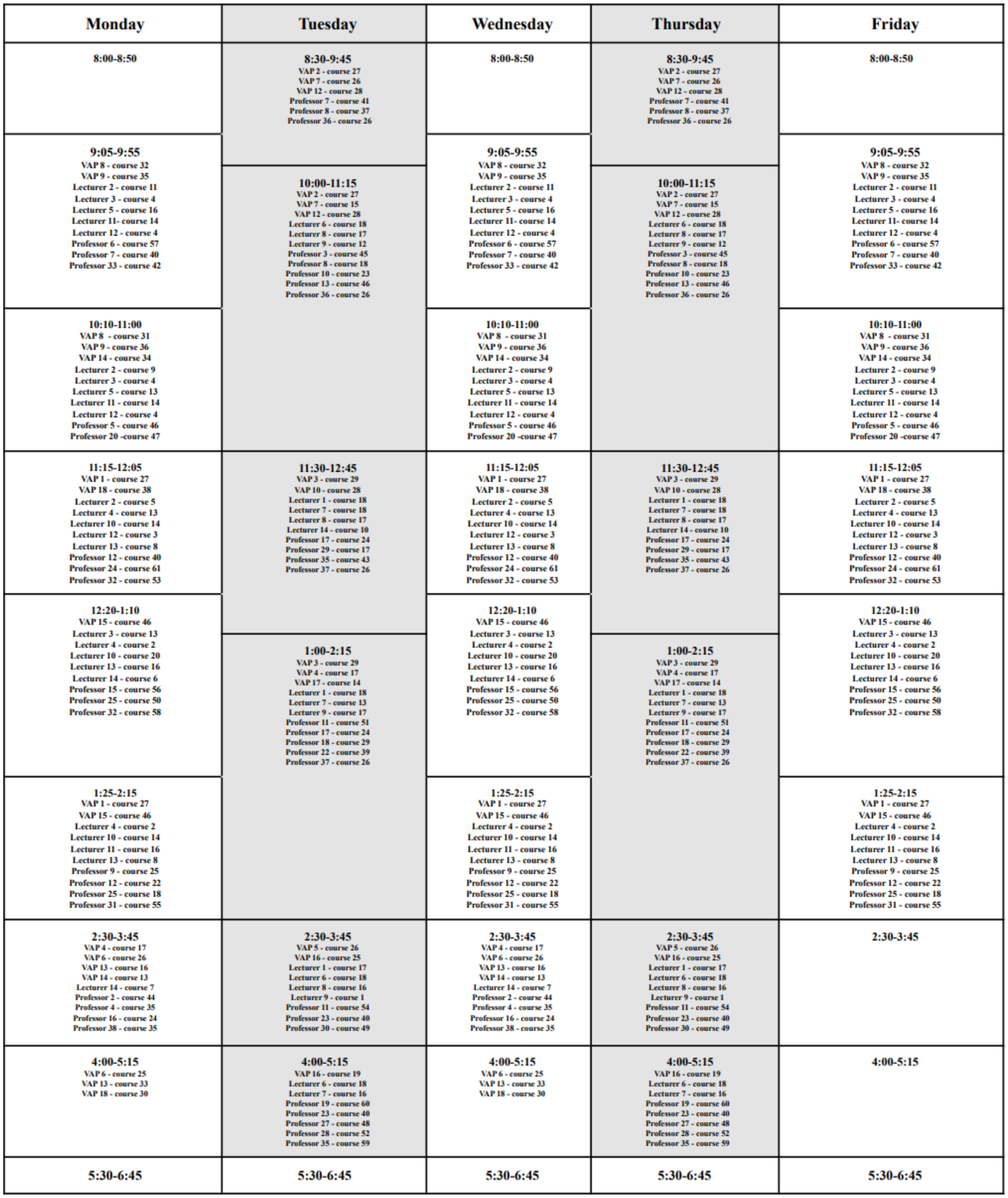
In the future, we would be interested in comparing our schedules to the one created by hand, and seeing where the main differences are. Getting feedback from instructors on how our schedules compare to those they have been given in the past in terms of their satisfaction with their assignments would also be helpful to know how well our program is working. There are a number of constraints that will be added in the future to account for more of the things the department currently considers when scheduling as well. Testing different formulations of the objective function for the schedule builder and different ways of synthesizing our current preference data into numerical weights may also give insight into ways to improve the program.

We hope the department is able to implement a version of a scheduling linear program to save time for employees that already have a larger-than-normal workload. We are confident that this is a process that can be automated, and that our current code is a good base that only needs relatively minor additions and changes to create schedules in line with the quality of those created by hand.

6 Bibliography

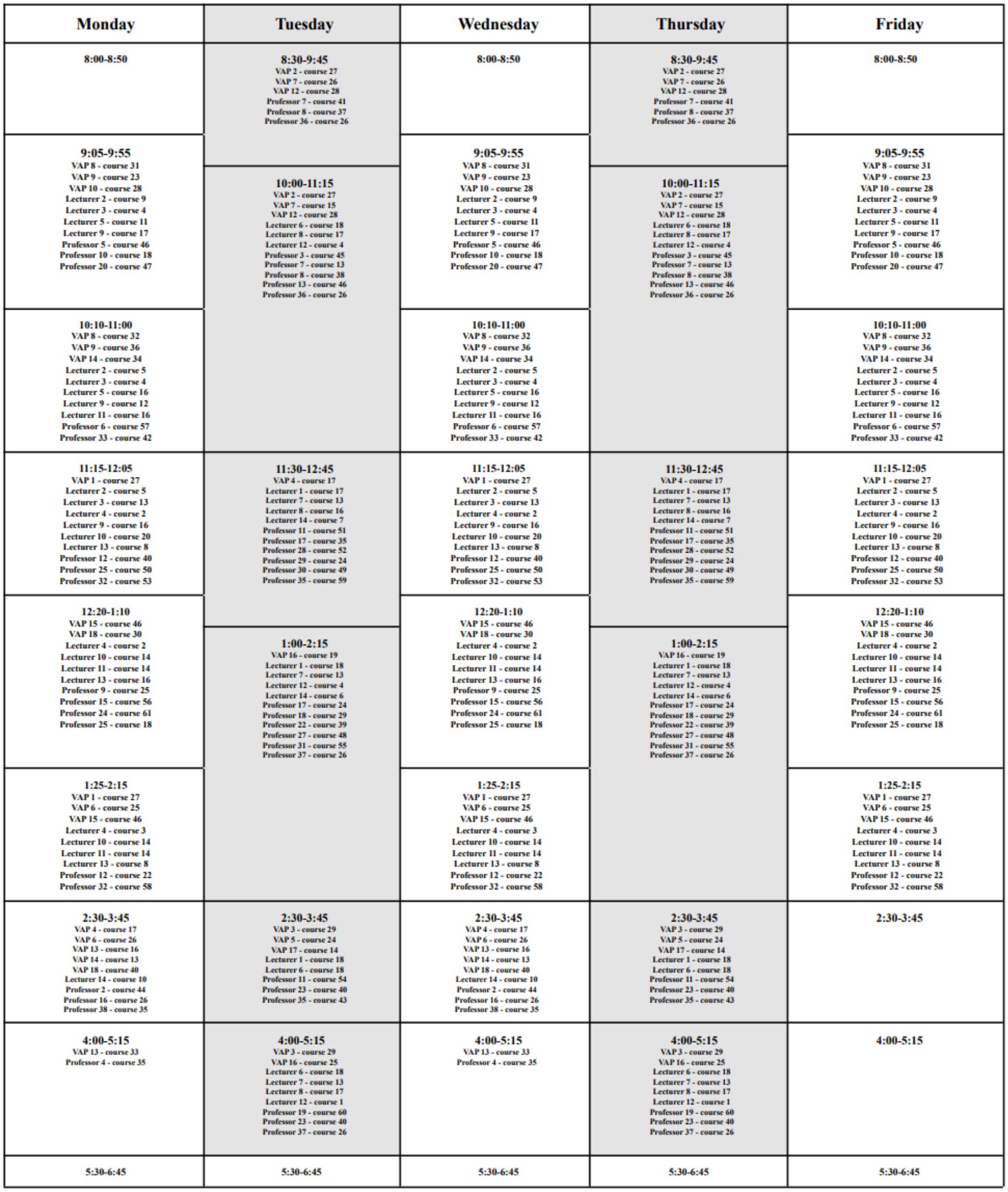
1. R. Kumar, "Modeling a Department Course Scheduling Problem Using Integer Progarmming: A Spreadsheet-Based Approach," Academy of Information and Management Sciences Journal, vol. 17, pp. 41–55, Nov. 2014.

2. M. C. Torres, K. K. S. Villegas, and M. K. A. Gavina, "Solving Faculty-Course Allocation Problem Using Integer Programming Model," Phillipine Journal of Science, vol. 150, no. 4, pp. 679–689, Aug. 2021.



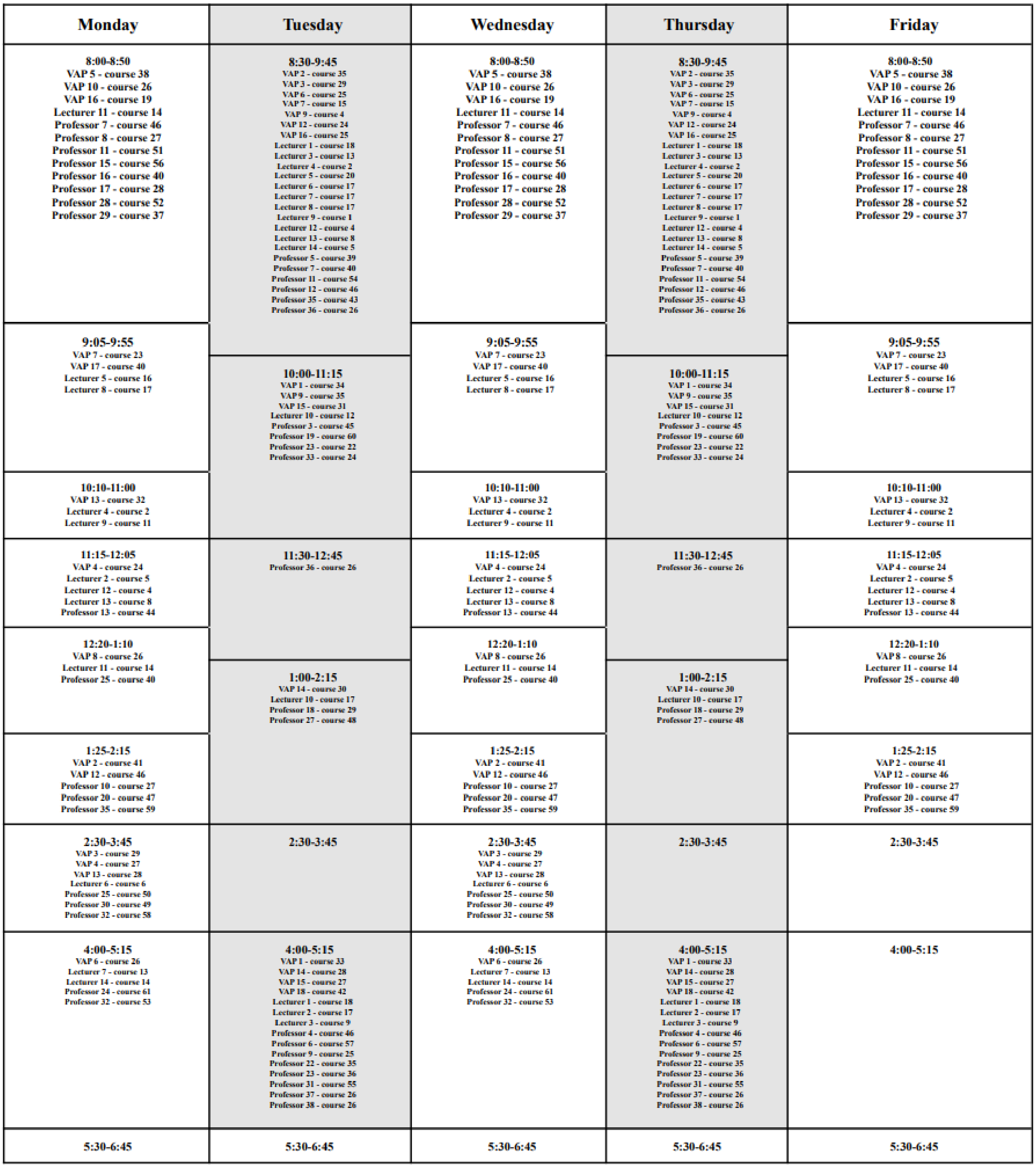
Monday	Tuesday	Wednesday	Thursday	Friday
8:00-8:50	**8:30-9:45** VAP 2 - course 27 VAP 7 - course 26 VAP 12 - course 28 Professor 7 - course 41 Professor 8 - course 37 Professor 36 - course 26	**8:00-8:50**	**8:30-9:45** VAP 2 - course 27 VAP 7 - course 26 VAP 12 - course 28 Professor 7 - course 41 Professor 8 - course 37 Professor 36 - course 26	**8:00-8:50**
9:05-9:55 VAP 8 - course 32 VAP 9 - course 35 Lecturer 2 - course 11 Lecturer 3 - course 4 Lecturer 5 - course 16 Lecturer 11- course 14 Lecturer 12 - course 4 Professor 6 - course 57 Professor 7 - course 40 Professor 33 - course 42	**10:00-11:15** VAP 2 - course 27 VAP 7 - course 15 VAP 12 - course 28 Lecturer 6 - course 18 Lecturer 8 - course 17 Lecturer 9 - course 12 Professor 3 - course 45 Professor 8 - course 18 Professor 10 - course 23 Professor 13 - course 46 Professor 36 - course 26	**9:05-9:55** VAP 8 - course 32 VAP 9 - course 35 Lecturer 2 - course 11 Lecturer 3 - course 4 Lecturer 5 - course 16 Lecturer 11- course 14 Lecturer 12 - course 4 Professor 6 - course 57 Professor 7 - course 40 Professor 33 - course 42	**10:00-11:15** VAP 2 - course 27 VAP 7 - course 15 VAP 12 - course 28 Lecturer 6 - course 18 Lecturer 8 - course 17 Lecturer 9 - course 12 Professor 3 - course 45 Professor 8 - course 18 Professor 10 - course 23 Professor 13 - course 46 Professor 36 - course 26	**9:05-9:55** VAP 8 - course 32 VAP 9 - course 35 Lecturer 2 - course 11 Lecturer 3 - course 4 Lecturer 5 - course 16 Lecturer 11- course 14 Lecturer 12 - course 4 Professor 6 - course 57 Professor 7 - course 40 Professor 33 - course 42
10:10-11:00 VAP 8 - course 31 VAP 9 - course 36 VAP 14 - course 34 Lecturer 2 - course 9 Lecturer 3 - course 4 Lecturer 5 - course 13 Lecturer 11 - course 14 Lecturer 12 - course 4 Professor 5 - course 46 Professor 20 -course 47		**10:10-11:00** VAP 8 - course 31 VAP 9 - course 36 VAP 14 - course 34 Lecturer 2 - course 9 Lecturer 3 - course 4 Lecturer 5 - course 13 Lecturer 11 - course 14 Lecturer 12 - course 4 Professor 5 - course 46 Professor 20 -course 47		**10:10-11:00** VAP 8 - course 31 VAP 9 - course 36 VAP 14 - course 34 Lecturer 2 - course 9 Lecturer 3 - course 4 Lecturer 5 - course 13 Lecturer 11 - course 14 Lecturer 12 - course 4 Professor 5 - course 46 Professor 20 -course 47
11:15-12:05 VAP 1 - course 27 VAP 18 - course 38 Lecturer 2 - course 5 Lecturer 4 - course 13 Lecturer 10 - course 14 Lecturer 12 - course 3 Lecturer 13 - course 8 Professor 12 - course 40 Professor 24 - course 61 Professor 32 - course 53	**11:30-12:45** VAP 3 - course 29 VAP 10 - course 28 Lecturer 1 - course 18 Lecturer 7 - course 18 Lecturer 8 - course 17 Lecturer 14 - course 10 Professor 17 - course 24 Professor 29 - course 17 Professor 35 - course 43 Professor 37 - course 26	**11:15-12:05** VAP 1 - course 27 VAP 18 - course 38 Lecturer 2 - course 5 Lecturer 4 - course 13 Lecturer 10 - course 14 Lecturer 12 - course 3 Lecturer 13 - course 8 Professor 12 - course 40 Professor 24 - course 61 Professor 32 - course 53	**11:30-12:45** VAP 3 - course 29 VAP 10 - course 28 Lecturer 1 - course 18 Lecturer 7 - course 18 Lecturer 8 - course 17 Lecturer 14 - course 10 Professor 17 - course 24 Professor 29 - course 17 Professor 35 - course 43 Professor 37 - course 26	**11:15-12:05** VAP 1 - course 27 VAP 18 - course 38 Lecturer 2 - course 5 Lecturer 4 - course 13 Lecturer 10 - course 14 Lecturer 12 - course 3 Lecturer 13 - course 8 Professor 12 - course 40 Professor 24 - course 61 Professor 32 - course 53
12:20-1:10 VAP 15 - course 46 Lecturer 3 - course 13 Lecturer 4 - course 2 Lecturer 10 - course 20 Lecturer 13 - course 16 Lecturer 14 - course 6 Professor 15 - course 56 Professor 25 - course 50 Professor 32 - course 58	**1:00-2:15** VAP 3 - course 29 VAP 4 - course 17 VAP 17 - course 14 Lecturer 1 - course 18 Lecturer 7 - course 13 Lecturer 9 - course 17 Professor 11 - course 51 Professor 17 - course 24 Professor 18 - course 29 Professor 22 - course 39 Professor 37 - course 26	**12:20-1:10** VAP 15 - course 46 Lecturer 3 - course 13 Lecturer 4 - course 2 Lecturer 10 - course 20 Lecturer 13 - course 16 Lecturer 14 - course 6 Professor 15 - course 56 Professor 25 - course 50 Professor 32 - course 58	**1:00-2:15** VAP 3 - course 29 VAP 4 - course 17 VAP 17 - course 14 Lecturer 1 - course 18 Lecturer 7 - course 13 Lecturer 9 - course 17 Professor 11 - course 51 Professor 17 - course 24 Professor 18 - course 29 Professor 22 - course 39 Professor 37 - course 26	**12:20-1:10** VAP 15 - course 46 Lecturer 3 - course 13 Lecturer 4 - course 2 Lecturer 10 - course 20 Lecturer 13 - course 16 Lecturer 14 - course 6 Professor 15 - course 56 Professor 25 - course 50 Professor 32 - course 58
1:25-2:15 VAP 1 - course 27 VAP 15 - course 46 Lecturer 4 - course 2 Lecturer 10 - course 14 Lecturer 11 - course 16 Lecturer 13 - course 8 Professor 9 - course 25 Professor 12 - course 22 Professor 25 - course 18 Professor 31 - course 55		**1:25-2:15** VAP 1 - course 27 VAP 15 - course 46 Lecturer 4 - course 2 Lecturer 10 - course 14 Lecturer 11 - course 16 Lecturer 13 - course 8 Professor 9 - course 25 Professor 12 - course 22 Professor 25 - course 18 Professor 31 - course 55		**1:25-2:15** VAP 1 - course 27 VAP 15 - course 46 Lecturer 4 - course 2 Lecturer 10 - course 14 Lecturer 11 - course 16 Lecturer 13 - course 8 Professor 9 - course 25 Professor 12 - course 22 Professor 25 - course 18 Professor 31 - course 55
2:30-3:45 VAP 4 - course 17 VAP 6 - course 26 VAP 13 - course 16 VAP 14 - course 13 Lecturer 14 - course 7 Professor 2 - course 44 Professor 4 - course 35 Professor 16 - course 24 Professor 38 - course 35	**2:30-3:45** VAP 5 - course 26 VAP 16 - course 25 Lecturer 1 - course 17 Lecturer 6 - course 18 Lecturer 8 - course 16 Lecturer 9 - course 1 Professor 11 - course 54 Professor 23 - course 40 Professor 30 - course 49	**2:30-3:45** VAP 4 - course 17 VAP 6 - course 26 VAP 13 - course 16 VAP 14 - course 13 Lecturer 14 - course 7 Professor 2 - course 44 Professor 4 - course 35 Professor 16 - course 24 Professor 38 - course 35	**2:30-3:45** VAP 5 - course 26 VAP 16 - course 25 Lecturer 1 - course 17 Lecturer 6 - course 18 Lecturer 8 - course 16 Lecturer 9 - course 1 Professor 11 - course 54 Professor 23 - course 40 Professor 30 - course 49	**2:30-3:45**
4:00-5:15 VAP 6 - course 25 VAP 13 - course 33 VAP 18 - course 30	**4:00-5:15** VAP 16 - course 19 Lecturer 6 - course 18 Lecturer 7 - course 16 Professor 19 - course 60 Professor 23 - course 40 Professor 27 - course 48 Professor 28 - course 52 Professor 35 - course 59	**4:00-5:15** VAP 6 - course 25 VAP 13 - course 33 VAP 18 - course 30	**4:00-5:15** VAP 16 - course 19 Lecturer 6 - course 18 Lecturer 7 - course 16 Professor 19 - course 60 Professor 23 - course 40 Professor 27 - course 48 Professor 28 - course 52 Professor 35 - course 59	**4:00-5:15**
5:30-6:45	**5:30-6:45**	**5:30-6:45**	**5:30-6:45**	**5:30-6:45**

Figure 3: Optimal Schedule 1 $(a * b)$



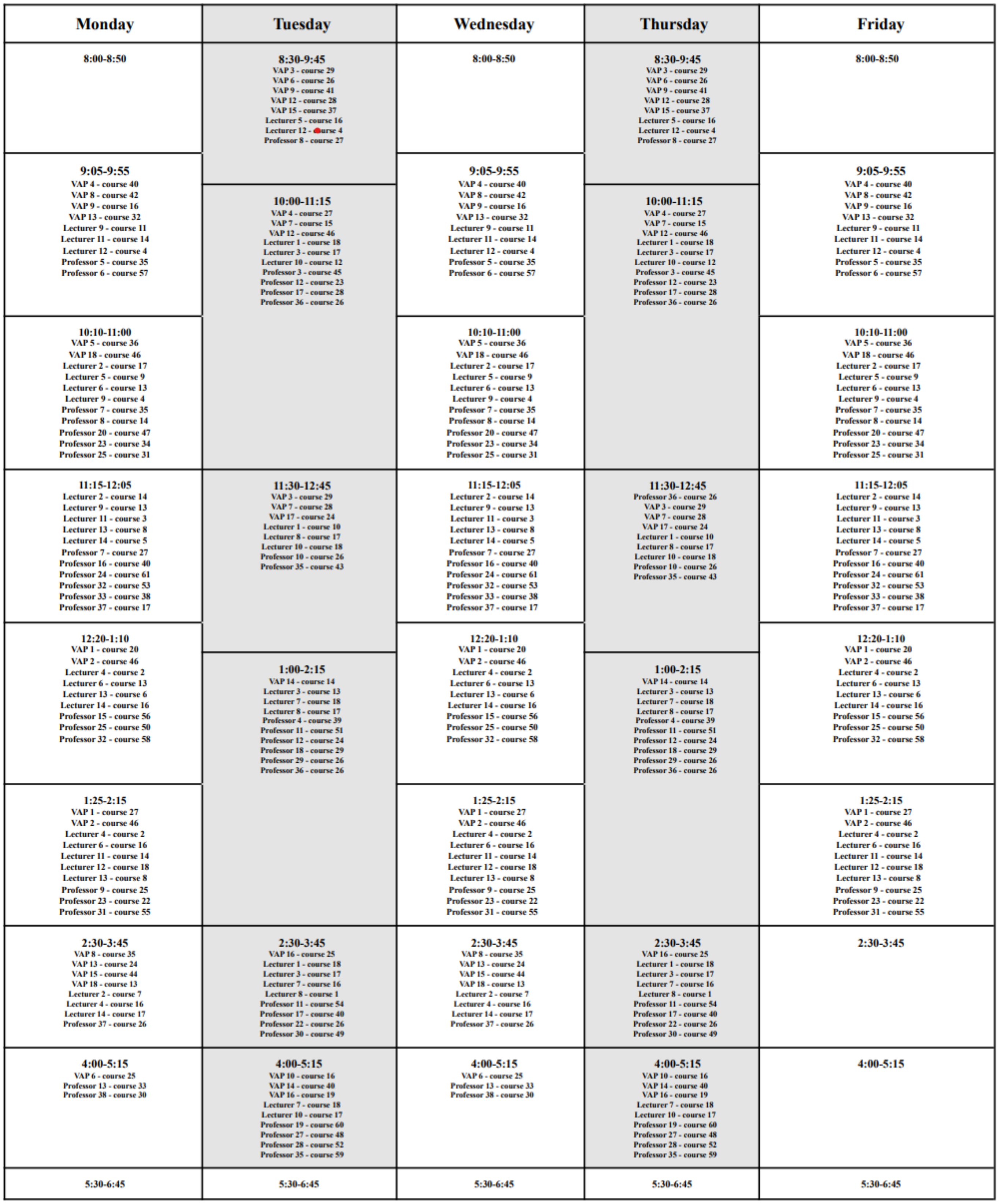
Monday	Tuesday	Wednesday	Thursday	Friday
8:00-8:50	8:30-9:45 VAP 2 - course 27 VAP 7 - course 26 VAP 12 - course 28 Professor 7 - course 41 Professor 8 - course 37 Professor 36 - course 26	8:00-8:50	8:30-9:45 VAP 2 - course 27 VAP 7 - course 26 VAP 12 - course 28 Professor 7 - course 41 Professor 8 - course 37 Professor 36 - course 26	8:00-8:50
9:05-9:55 VAP 8 - course 31 VAP 9 - course 23 VAP 10 - course 28 Lecturer 2 - course 9 Lecturer 3 - course 4 Lecturer 5 - course 11 Lecturer 9 - course 17 Professor 5 - course 46 Professor 10 - course 18 Professor 20 - course 47	10:00-11:15 VAP 2 - course 27 VAP 7 - course 15 VAP 12 - course 28 Lecturer 6 - course 18 Lecturer 8 - course 17 Lecturer 12 - course 4 Professor 3 - course 45 Professor 7 - course 13 Professor 8 - course 38 Professor 13 - course 46 Professor 36 - course 26	9:05-9:55 VAP 8 - course 31 VAP 9 - course 23 VAP 10 - course 28 Lecturer 2 - course 9 Lecturer 3 - course 4 Lecturer 5 - course 11 Lecturer 9 - course 17 Professor 5 - course 46 Professor 10 - course 18 Professor 20 - course 47	10:00-11:15 VAP 2 - course 27 VAP 7 - course 15 VAP 12 - course 28 Lecturer 6 - course 18 Lecturer 8 - course 17 Lecturer 12 - course 4 Professor 3 - course 45 Professor 7 - course 13 Professor 8 - course 38 Professor 13 - course 46 Professor 36 - course 26	9:05-9:55 VAP 8 - course 31 VAP 9 - course 23 VAP 10 - course 28 Lecturer 2 - course 9 Lecturer 3 - course 4 Lecturer 5 - course 11 Lecturer 9 - course 17 Professor 5 - course 46 Professor 10 - course 18 Professor 20 - course 47
10:10-11:00 VAP 8 - course 32 VAP 9 - course 36 VAP 14 - course 34 Lecturer 2 - course 5 Lecturer 3 - course 4 Lecturer 5 - course 16 Lecturer 9 - course 12 Lecturer 11 - course 16 Professor 6 - course 57 Professor 33 - course 42		10:10-11:00 VAP 8 - course 32 VAP 9 - course 36 VAP 14 - course 34 Lecturer 2 - course 5 Lecturer 3 - course 4 Lecturer 5 - course 16 Lecturer 9 - course 12 Lecturer 11 - course 16 Professor 6 - course 57 Professor 33 - course 42		10:10-11:00 VAP 8 - course 32 VAP 9 - course 36 VAP 14 - course 34 Lecturer 2 - course 5 Lecturer 3 - course 4 Lecturer 5 - course 16 Lecturer 9 - course 12 Lecturer 11 - course 16 Professor 6 - course 57 Professor 33 - course 42
11:15-12:05 VAP 1 - course 27 Lecturer 2 - course 5 Lecturer 3 - course 13 Lecturer 4 - course 2 Lecturer 9 - course 16 Lecturer 10 - course 20 Lecturer 13 - course 8 Professor 12 - course 40 Professor 25 - course 50 Professor 32 - course 53	11:30-12:45 VAP 4 - course 17 Lecturer 1 - course 17 Lecturer 7 - course 13 Lecturer 8 - course 16 Lecturer 14 - course 7 Professor 11 - course 51 Professor 17 - course 35 Professor 28 - course 52 Professor 29 - course 24 Professor 30 - course 49 Professor 35 - course 59	11:15-12:05 VAP 1 - course 27 Lecturer 2 - course 5 Lecturer 3 - course 13 Lecturer 4 - course 2 Lecturer 9 - course 16 Lecturer 10 - course 20 Lecturer 13 - course 8 Professor 12 - course 40 Professor 25 - course 50 Professor 32 - course 53	11:30-12:45 VAP 4 - course 17 Lecturer 1 - course 17 Lecturer 7 - course 13 Lecturer 8 - course 16 Lecturer 14 - course 7 Professor 11 - course 51 Professor 17 - course 35 Professor 28 - course 52 Professor 29 - course 24 Professor 30 - course 49 Professor 35 - course 59	11:15-12:05 VAP 1 - course 27 Lecturer 2 - course 5 Lecturer 3 - course 13 Lecturer 4 - course 2 Lecturer 9 - course 16 Lecturer 10 - course 20 Lecturer 13 - course 8 Professor 12 - course 40 Professor 25 - course 50 Professor 32 - course 53
12:20-1:10 VAP 15 - course 46 VAP 18 - course 30 Lecturer 4 - course 2 Lecturer 10 - course 14 Lecturer 11 - course 14 Lecturer 13 - course 16 Professor 9 - course 25 Professor 15 - course 56 Professor 24 - course 61 Professor 25 - course 18	1:00-2:15 VAP 16 - course 19 Lecturer 1 - course 18 Lecturer 7 - course 13 Lecturer 12 - course 4 Lecturer 14 - course 6 Professor 17 - course 24 Professor 18 - course 29 Professor 22 - course 39 Professor 27 - course 48 Professor 31 - course 55 Professor 37 - course 26	12:20-1:10 VAP 15 - course 46 VAP 18 - course 30 Lecturer 4 - course 2 Lecturer 10 - course 14 Lecturer 11 - course 14 Lecturer 13 - course 16 Professor 9 - course 25 Professor 15 - course 56 Professor 24 - course 61 Professor 25 - course 18	1:00-2:15 VAP 16 - course 19 Lecturer 1 - course 18 Lecturer 7 - course 13 Lecturer 12 - course 4 Lecturer 14 - course 6 Professor 17 - course 24 Professor 18 - course 29 Professor 22 - course 39 Professor 27 - course 48 Professor 31 - course 55 Professor 37 - course 26	12:20-1:10 VAP 15 - course 46 VAP 18 - course 30 Lecturer 4 - course 2 Lecturer 10 - course 14 Lecturer 11 - course 14 Lecturer 13 - course 16 Professor 9 - course 25 Professor 15 - course 56 Professor 24 - course 61 Professor 25 - course 18
1:25-2:15 VAP 1 - course 27 VAP 6 - course 25 VAP 15 - course 46 Lecturer 4 - course 3 Lecturer 10 - course 14 Lecturer 11 - course 14 Lecturer 13 - course 8 Professor 12 - course 22 Professor 32 - course 58		1:25-2:15 VAP 1 - course 27 VAP 6 - course 25 VAP 15 - course 46 Lecturer 4 - course 3 Lecturer 10 - course 14 Lecturer 11 - course 14 Lecturer 13 - course 8 Professor 12 - course 22 Professor 32 - course 58		1:25-2:15 VAP 1 - course 27 VAP 6 - course 25 VAP 15 - course 46 Lecturer 4 - course 3 Lecturer 10 - course 14 Lecturer 11 - course 14 Lecturer 13 - course 8 Professor 12 - course 22 Professor 32 - course 58
2:30-3:45 VAP 4 - course 17 VAP 6 - course 26 VAP 13 - course 16 VAP 14 - course 13 VAP 18 - course 40 Lecturer 14 - course 10 Professor 2 - course 44 Professor 16 - course 26 Professor 38 - course 35	2:30-3:45 VAP 3 - course 29 VAP 5 - course 24 VAP 17 - course 14 Lecturer 1 - course 18 Lecturer 6 - course 18 Professor 11 - course 54 Professor 23 - course 40 Professor 35 - course 43	2:30-3:45 VAP 4 - course 17 VAP 6 - course 26 VAP 13 - course 16 VAP 14 - course 13 VAP 18 - course 40 Lecturer 14 - course 10 Professor 2 - course 44 Professor 16 - course 26 Professor 38 - course 35	2:30-3:45 VAP 3 - course 29 VAP 5 - course 24 VAP 17 - course 14 Lecturer 1 - course 18 Lecturer 6 - course 18 Professor 11 - course 54 Professor 23 - course 40 Professor 35 - course 43	2:30-3:45
4:00-5:15 VAP 13 - course 33 Professor 4 - course 35	4:00-5:15 VAP 3 - course 29 VAP 16 - course 25 Lecturer 6 - course 18 Lecturer 7 - course 13 Lecturer 8 - course 17 Lecturer 12 - course 1 Professor 19 - course 60 Professor 23 - course 40 Professor 37 - course 26	4:00-5:15 VAP 13 - course 33 Professor 4 - course 35	4:00-5:15 VAP 3 - course 29 VAP 16 - course 25 Lecturer 6 - course 18 Lecturer 7 - course 13 Lecturer 8 - course 17 Lecturer 12 - course 1 Professor 19 - course 60 Professor 23 - course 40 Professor 37 - course 26	4:00-5:15
5:30-6:45	5:30-6:45	5:30-6:45	5:30-6:45	5:30-6:45

Figure 4: Optimal Schedule 2 $(a + b)$



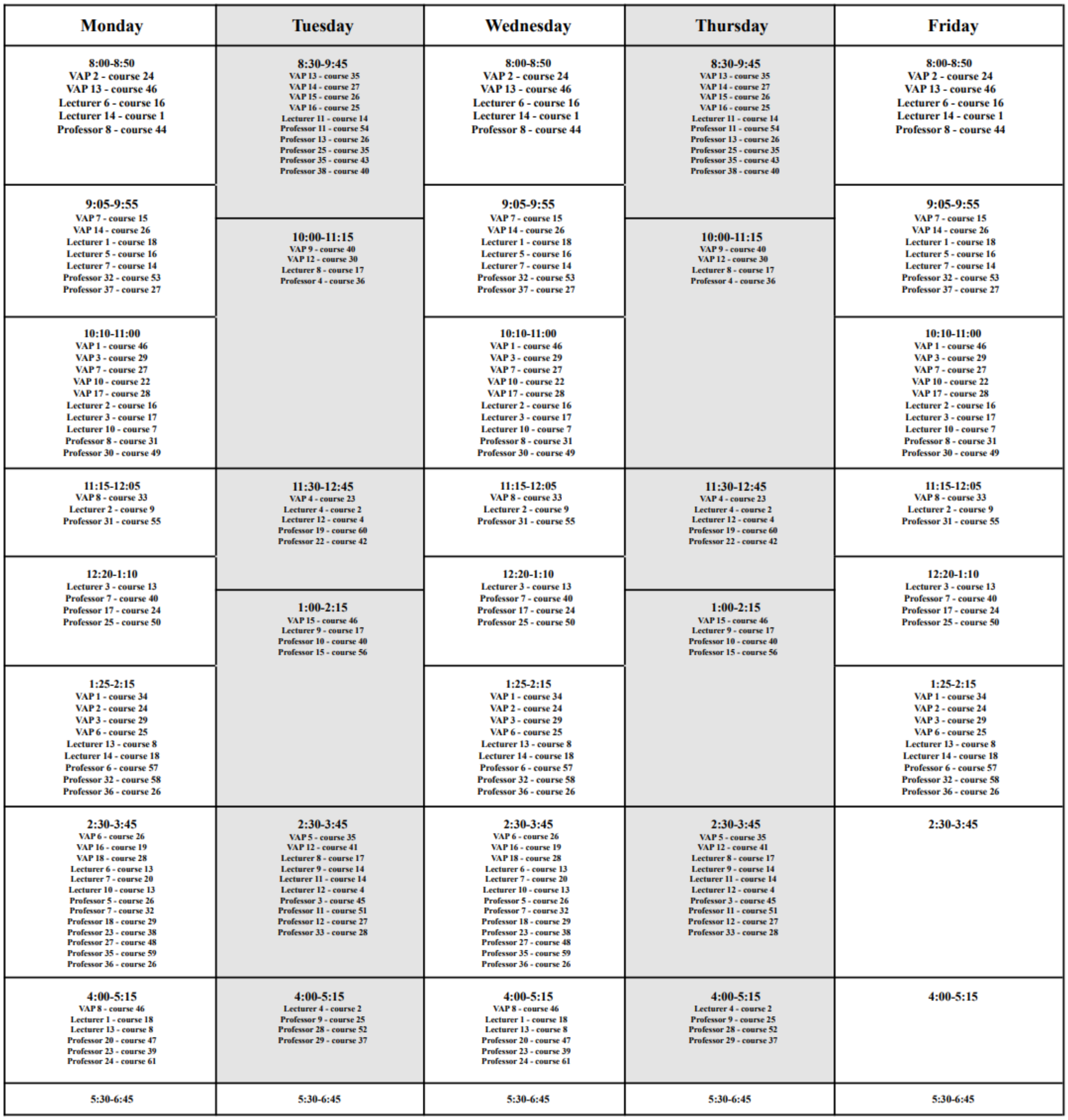
Monday	Tuesday	Wednesday	Thursday	Friday
8:00-8:50; VAP 5 - course 38; VAP 10 - course 26; VAP 16 - course 19; Lecturer 11 - course 14; Professor 7 - course 46; Professor 8 - course 27; Professor 11 - course 51; Professor 15 - course 56; Professor 16 - course 40; Professor 17 - course 28; Professor 28 - course 52; Professor 29 - course 37	8:30-9:45; VAP 2 - course 35; VAP 3 - course 29; VAP 6 - course 25; VAP 7 - course 15; VAP 9 - course 4; VAP 12 - course 24; VAP 16 - course 25; Lecturer 1 - course 18; Lecturer 3 - course 13; Lecturer 4 - course 2; Lecturer 5 - course 20; Lecturer 6 - course 17; Lecturer 7 - course 17; Lecturer 8 - course 17; Lecturer 9 - course 1; Lecturer 12 - course 4; Lecturer 13 - course 8; Lecturer 14 - course 5; Professor 5 - course 39; Professor 7 - course 40; Professor 11 - course 54; Professor 12 - course 46; Professor 35 - course 43; Professor 36 - course 26	8:00-8:50; VAP 5 - course 38; VAP 10 - course 26; VAP 16 - course 19; Lecturer 11 - course 14; Professor 7 - course 46; Professor 8 - course 27; Professor 11 - course 51; Professor 15 - course 56; Professor 16 - course 40; Professor 17 - course 28; Professor 28 - course 52; Professor 29 - course 37	8:30-9:45; VAP 2 - course 35; VAP 3 - course 29; VAP 6 - course 25; VAP 7 - course 15; VAP 9 - course 4; VAP 12 - course 24; VAP 16 - course 25; Lecturer 1 - course 18; Lecturer 3 - course 13; Lecturer 4 - course 2; Lecturer 5 - course 20; Lecturer 6 - course 17; Lecturer 7 - course 17; Lecturer 8 - course 17; Lecturer 9 - course 1; Lecturer 12 - course 4; Lecturer 13 - course 8; Lecturer 14 - course 5; Professor 5 - course 39; Professor 7 - course 40; Professor 11 - course 54; Professor 12 - course 46; Professor 35 - course 43; Professor 36 - course 26	8:00-8:50; VAP 5 - course 38; VAP 10 - course 26; VAP 16 - course 19; Lecturer 11 - course 14; Professor 7 - course 46; Professor 8 - course 27; Professor 11 - course 51; Professor 15 - course 56; Professor 16 - course 40; Professor 17 - course 28; Professor 28 - course 52; Professor 29 - course 37
9:05-9:55; VAP 7 - course 23; VAP 17 - course 40; Lecturer 5 - course 16; Lecturer 8 - course 17	10:00-11:15; VAP 1 - course 34; VAP 9 - course 35; VAP 15 - course 31; Lecturer 10 - course 12; Professor 3 - course 45; Professor 19 - course 60; Professor 23 - course 22; Professor 33 - course 24	9:05-9:55; VAP 7 - course 23; VAP 17 - course 40; Lecturer 5 - course 16; Lecturer 8 - course 17	10:00-11:15; VAP 1 - course 34; VAP 9 - course 35; VAP 15 - course 31; Lecturer 10 - course 12; Professor 3 - course 45; Professor 19 - course 60; Professor 23 - course 22; Professor 33 - course 24	9:05-9:55; VAP 7 - course 23; VAP 17 - course 40; Lecturer 5 - course 16; Lecturer 8 - course 17
10:10-11:00; VAP 13 - course 32; Lecturer 4 - course 2; Lecturer 9 - course 11		10:10-11:00; VAP 13 - course 32; Lecturer 4 - course 2; Lecturer 9 - course 11		10:10-11:00; VAP 13 - course 32; Lecturer 4 - course 2; Lecturer 9 - course 11
11:15-12:05; VAP 4 - course 24; Lecturer 2 - course 5; Lecturer 12 - course 4; Lecturer 13 - course 8; Professor 13 - course 44	11:30-12:45; Professor 36 - course 26	11:15-12:05; VAP 4 - course 24; Lecturer 2 - course 5; Lecturer 12 - course 4; Lecturer 13 - course 8; Professor 13 - course 44	11:30-12:45; Professor 36 - course 26	11:15-12:05; VAP 4 - course 24; Lecturer 2 - course 5; Lecturer 12 - course 4; Lecturer 13 - course 8; Professor 13 - course 44
12:20-1:10; VAP 8 - course 26; Lecturer 11 - course 14; Professor 25 - course 40	1:00-2:15; VAP 14 - course 30; Lecturer 10 - course 17; Professor 18 - course 29; Professor 27 - course 48	12:20-1:10; VAP 8 - course 26; Lecturer 11 - course 14; Professor 25 - course 40	1:00-2:15; VAP 14 - course 30; Lecturer 10 - course 17; Professor 18 - course 29; Professor 27 - course 48	12:20-1:10; VAP 8 - course 26; Lecturer 11 - course 14; Professor 25 - course 40
1:25-2:15; VAP 2 - course 41; VAP 12 - course 46; Professor 10 - course 27; Professor 20 - course 47; Professor 35 - course 59		1:25-2:15; VAP 2 - course 41; VAP 12 - course 46; Professor 10 - course 27; Professor 20 - course 47; Professor 35 - course 59		1:25-2:15; VAP 2 - course 41; VAP 12 - course 46; Professor 10 - course 27; Professor 20 - course 47; Professor 35 - course 59
2:30-3:45; VAP 3 - course 29; VAP 4 - course 27; VAP 13 - course 28; Lecturer 6 - course 6; Professor 25 - course 50; Professor 30 - course 49; Professor 32 - course 58	2:30-3:45	2:30-3:45; VAP 3 - course 29; VAP 4 - course 27; VAP 13 - course 28; Lecturer 6 - course 6; Professor 25 - course 50; Professor 30 - course 49; Professor 32 - course 58	2:30-3:45	2:30-3:45
4:00-5:15; VAP 6 - course 26; Lecturer 7 - course 13; Lecturer 14 - course 14; Professor 24 - course 61; Professor 32 - course 53	4:00-5:15; VAP 1 - course 33; VAP 14 - course 28; VAP 15 - course 27; VAP 18 - course 42; Lecturer 1 - course 18; Lecturer 2 - course 17; Lecturer 3 - course 9; Professor 4 - course 46; Professor 6 - course 57; Professor 9 - course 25; Professor 22 - course 35; Professor 23 - course 36; Professor 31 - course 55; Professor 37 - course 26; Professor 38 - course 26	4:00-5:15; VAP 6 - course 26; Lecturer 7 - course 13; Lecturer 14 - course 14; Professor 24 - course 61; Professor 32 - course 53	4:00-5:15; VAP 1 - course 33; VAP 14 - course 28; VAP 15 - course 27; VAP 18 - course 42; Lecturer 1 - course 18; Lecturer 2 - course 17; Lecturer 3 - course 9; Professor 4 - course 46; Professor 6 - course 57; Professor 9 - course 25; Professor 22 - course 35; Professor 23 - course 36; Professor 31 - course 55; Professor 37 - course 26; Professor 38 - course 26	4:00-5:15
5:30-6:45	5:30-6:45	5:30-6:45	5:30-6:45	5:30-6:45

Figure 5: Schedule 1 Pre-Registration Changes



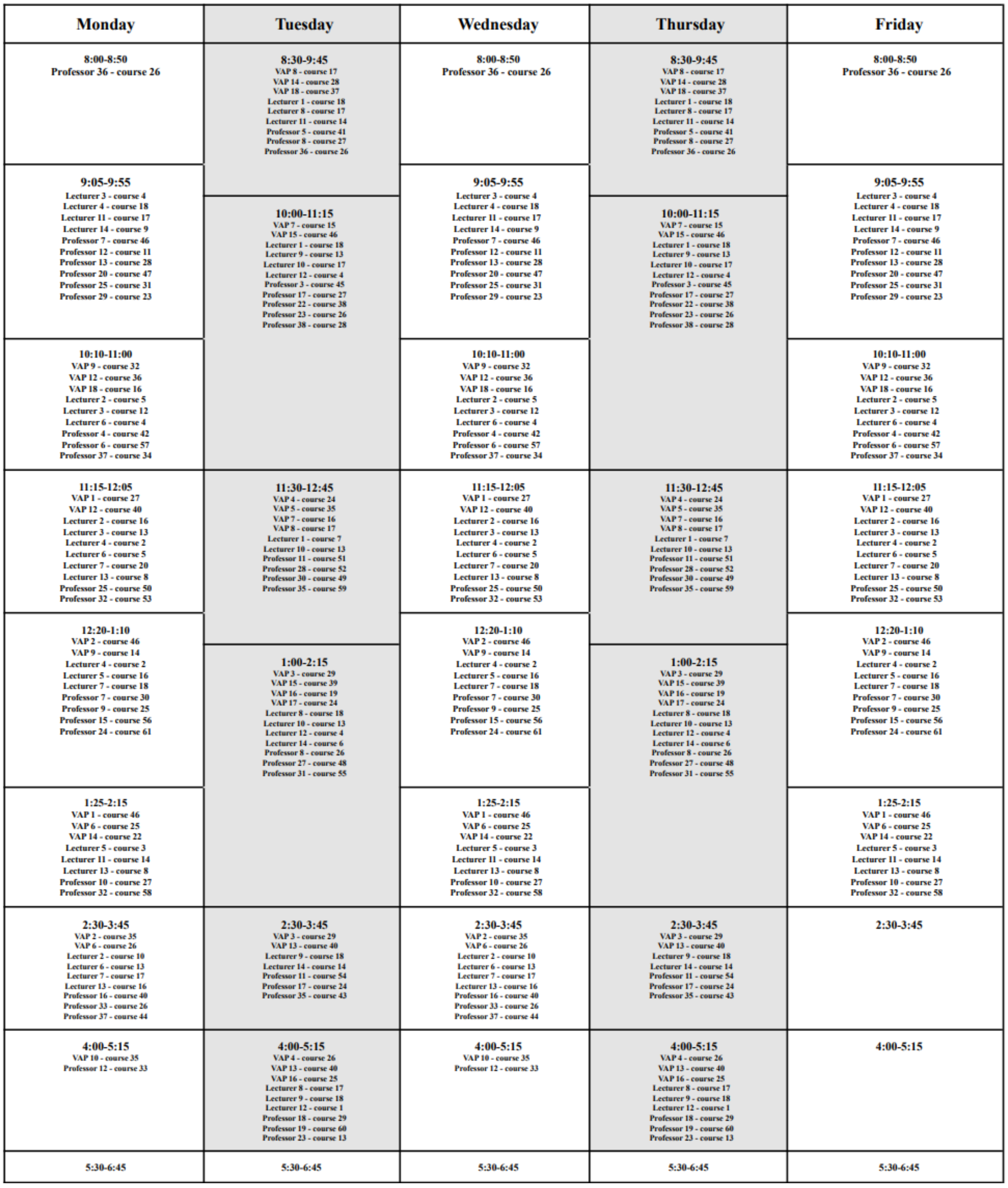
Monday	Tuesday	Wednesday	Thursday	Friday
8:00-8:50	**8:30-9:45** VAP 3 - course 29 VAP 6 - course 26 VAP 9 - course 41 VAP 12 - course 28 VAP 15 - course 37 Lecturer 5 - course 16 Lecturer 12 - course 4 Professor 8 - course 27	**8:00-8:50**	**8:30-9:45** VAP 3 - course 29 VAP 6 - course 26 VAP 9 - course 41 VAP 12 - course 28 VAP 15 - course 37 Lecturer 5 - course 16 Lecturer 12 - course 4 Professor 8 - course 27	**8:00-8:50**
9:05-9:55 VAP 4 - course 40 VAP 8 - course 42 VAP 9 - course 16 VAP 13 - course 32 Lecturer 9 - course 11 Lecturer 11 - course 14 Lecturer 12 - course 4 Professor 5 - course 35 Professor 6 - course 57	**10:00-11:15** VAP 4 - course 27 VAP 7 - course 15 VAP 12 - course 46 Lecturer 1 - course 18 Lecturer 3 - course 17 Lecturer 10 - course 12 Professor 3 - course 45 Professor 12 - course 23 Professor 17 - course 28 Professor 36 - course 26	**9:05-9:55** VAP 4 - course 40 VAP 8 - course 42 VAP 9 - course 16 VAP 13 - course 32 Lecturer 9 - course 11 Lecturer 11 - course 14 Lecturer 12 - course 4 Professor 5 - course 35 Professor 6 - course 57	**10:00-11:15** VAP 4 - course 27 VAP 7 - course 15 VAP 12 - course 46 Lecturer 1 - course 18 Lecturer 3 - course 17 Lecturer 10 - course 12 Professor 3 - course 45 Professor 12 - course 23 Professor 17 - course 28 Professor 36 - course 26	**9:05-9:55** VAP 4 - course 40 VAP 8 - course 42 VAP 9 - course 16 VAP 13 - course 32 Lecturer 9 - course 11 Lecturer 11 - course 14 Lecturer 12 - course 4 Professor 5 - course 35 Professor 6 - course 57
10:10-11:00 VAP 5 - course 36 VAP 18 - course 46 Lecturer 2 - course 17 Lecturer 5 - course 9 Lecturer 6 - course 13 Lecturer 9 - course 4 Professor 7 - course 35 Professor 8 - course 14 Professor 20 - course 47 Professor 23 - course 34 Professor 25 - course 31		**10:10-11:00** VAP 5 - course 36 VAP 18 - course 46 Lecturer 2 - course 17 Lecturer 5 - course 9 Lecturer 6 - course 13 Lecturer 9 - course 4 Professor 7 - course 35 Professor 8 - course 14 Professor 20 - course 47 Professor 23 - course 34 Professor 25 - course 31		**10:10-11:00** VAP 5 - course 36 VAP 18 - course 46 Lecturer 2 - course 17 Lecturer 5 - course 9 Lecturer 6 - course 13 Lecturer 9 - course 4 Professor 7 - course 35 Professor 8 - course 14 Professor 20 - course 47 Professor 23 - course 34 Professor 25 - course 31
11:15-12:05 Lecturer 2 - course 14 Lecturer 9 - course 13 Lecturer 11 - course 3 Lecturer 13 - course 8 Lecturer 14 - course 5 Professor 7 - course 27 Professor 16 - course 40 Professor 24 - course 61 Professor 32 - course 53 Professor 33 - course 38 Professor 37 - course 17	**11:30-12:45** VAP 3 - course 29 VAP 7 - course 28 VAP 17 - course 24 Lecturer 1 - course 10 Lecturer 8 - course 17 Lecturer 10 - course 18 Professor 10 - course 26 Professor 35 - course 43	**11:15-12:05** Lecturer 2 - course 14 Lecturer 9 - course 13 Lecturer 11 - course 3 Lecturer 13 - course 8 Lecturer 14 - course 5 Professor 7 - course 27 Professor 16 - course 40 Professor 24 - course 61 Professor 32 - course 53 Professor 33 - course 38 Professor 37 - course 17	**11:30-12:45** Professor 36 - course 26 VAP 3 - course 29 VAP 7 - course 28 VAP 17 - course 24 Lecturer 1 - course 10 Lecturer 8 - course 17 Lecturer 10 - course 18 Professor 10 - course 26 Professor 35 - course 43	**11:15-12:05** Lecturer 2 - course 14 Lecturer 9 - course 13 Lecturer 11 - course 3 Lecturer 13 - course 8 Lecturer 14 - course 5 Professor 7 - course 27 Professor 16 - course 40 Professor 24 - course 61 Professor 32 - course 53 Professor 33 - course 38 Professor 37 - course 17
12:20-1:10 VAP 1 - course 20 VAP 2 - course 46 Lecturer 4 - course 2 Lecturer 6 - course 13 Lecturer 13 - course 6 Lecturer 14 - course 16 Professor 15 - course 56 Professor 25 - course 50 Professor 32 - course 58	**1:00-2:15** VAP 14 - course 14 Lecturer 3 - course 13 Lecturer 7 - course 18 Lecturer 8 - course 17 Professor 4 - course 39 Professor 11 - course 51 Professor 12 - course 24 Professor 18 - course 29 Professor 29 - course 26 Professor 36 - course 26	**12:20-1:10** VAP 1 - course 20 VAP 2 - course 46 Lecturer 4 - course 2 Lecturer 6 - course 13 Lecturer 13 - course 6 Lecturer 14 - course 16 Professor 15 - course 56 Professor 25 - course 50 Professor 32 - course 58	**1:00-2:15** VAP 14 - course 14 Lecturer 3 - course 13 Lecturer 7 - course 18 Lecturer 8 - course 17 Professor 4 - course 39 Professor 11 - course 51 Professor 12 - course 24 Professor 18 - course 29 Professor 29 - course 26 Professor 36 - course 26	**12:20-1:10** VAP 1 - course 20 VAP 2 - course 46 Lecturer 4 - course 2 Lecturer 6 - course 13 Lecturer 13 - course 6 Lecturer 14 - course 16 Professor 15 - course 56 Professor 25 - course 50 Professor 32 - course 58
1:25-2:15 VAP 1 - course 27 VAP 2 - course 46 Lecturer 4 - course 2 Lecturer 6 - course 16 Lecturer 11 - course 14 Lecturer 12 - course 18 Lecturer 13 - course 8 Professor 9 - course 25 Professor 23 - course 22 Professor 31 - course 55		**1:25-2:15** VAP 1 - course 27 VAP 2 - course 46 Lecturer 4 - course 2 Lecturer 6 - course 16 Lecturer 11 - course 14 Lecturer 12 - course 18 Lecturer 13 - course 8 Professor 9 - course 25 Professor 23 - course 22 Professor 31 - course 55		**1:25-2:15** VAP 1 - course 27 VAP 2 - course 46 Lecturer 4 - course 2 Lecturer 6 - course 16 Lecturer 11 - course 14 Lecturer 12 - course 18 Lecturer 13 - course 8 Professor 9 - course 25 Professor 23 - course 22 Professor 31 - course 55
2:30-3:45 VAP 8 - course 35 VAP 13 - course 24 VAP 15 - course 44 VAP 18 - course 13 Lecturer 2 - course 7 Lecturer 4 - course 16 Lecturer 14 - course 17 Professor 37 - course 26	**2:30-3:45** VAP 16 - course 25 Lecturer 1 - course 18 Lecturer 3 - course 17 Lecturer 7 - course 16 Lecturer 8 - course 1 Professor 11 - course 54 Professor 17 - course 40 Professor 22 - course 26 Professor 30 - course 49	**2:30-3:45** VAP 8 - course 35 VAP 13 - course 24 VAP 15 - course 44 VAP 18 - course 13 Lecturer 2 - course 7 Lecturer 4 - course 16 Lecturer 14 - course 17 Professor 37 - course 26	**2:30-3:45** VAP 16 - course 25 Lecturer 1 - course 18 Lecturer 3 - course 17 Lecturer 7 - course 16 Lecturer 8 - course 1 Professor 11 - course 54 Professor 17 - course 40 Professor 22 - course 26 Professor 30 - course 49	**2:30-3:45**
4:00-5:15 VAP 6 - course 25 Professor 13 - course 33 Professor 38 - course 30	**4:00-5:15** VAP 10 - course 16 VAP 14 - course 40 VAP 16 - course 19 Lecturer 7 - course 18 Lecturer 10 - course 17 Professor 19 - course 60 Professor 27 - course 48 Professor 28 - course 52 Professor 35 - course 59	**4:00-5:15** VAP 6 - course 25 Professor 13 - course 33 Professor 38 - course 30	**4:00-5:15** VAP 10 - course 16 VAP 14 - course 40 VAP 16 - course 19 Lecturer 7 - course 18 Lecturer 10 - course 17 Professor 19 - course 60 Professor 27 - course 48 Professor 28 - course 52 Professor 35 - course 59	**4:00-5:15**
5:30-6:45	**5:30-6:45**	**5:30-6:45**	**5:30-6:45**	**5:30-6:45**

Figure 6: Schedule 1 Post-Registration Changes



Monday	Tuesday	Wednesday	Thursday	Friday
8:00-8:50 VAP 2 - course 24 VAP 13 - course 46 Lecturer 6 - course 16 Lecturer 14 - course 1 Professor 8 - course 44	8:30-9:45 VAP 13 - course 35 VAP 14 - course 27 VAP 15 - course 26 VAP 16 - course 25 Lecturer 11 - course 14 Professor 11 - course 54 Professor 13 - course 26 Professor 25 - course 35 Professor 35 - course 43 Professor 38 - course 40	8:00-8:50 VAP 2 - course 24 VAP 13 - course 46 Lecturer 6 - course 16 Lecturer 14 - course 1 Professor 8 - course 44	8:30-9:45 VAP 13 - course 35 VAP 14 - course 27 VAP 15 - course 26 VAP 16 - course 25 Lecturer 11 - course 14 Professor 11 - course 54 Professor 13 - course 26 Professor 25 - course 35 Professor 35 - course 43 Professor 38 - course 40	8:00-8:50 VAP 2 - course 24 VAP 13 - course 46 Lecturer 6 - course 16 Lecturer 14 - course 1 Professor 8 - course 44
9:05-9:55 VAP 7 - course 15 VAP 14 - course 26 Lecturer 1 - course 18 Lecturer 5 - course 16 Lecturer 7 - course 14 Professor 32 - course 53 Professor 37 - course 27	10:00-11:15 VAP 9 - course 40 VAP 12 - course 30 Lecturer 8 - course 17 Professor 4 - course 36	9:05-9:55 VAP 7 - course 15 VAP 14 - course 26 Lecturer 1 - course 18 Lecturer 5 - course 16 Lecturer 7 - course 14 Professor 32 - course 53 Professor 37 - course 27	10:00-11:15 VAP 9 - course 40 VAP 12 - course 30 Lecturer 8 - course 17 Professor 4 - course 36	9:05-9:55 VAP 7 - course 15 VAP 14 - course 26 Lecturer 1 - course 18 Lecturer 5 - course 16 Lecturer 7 - course 14 Professor 32 - course 53 Professor 37 - course 27
10:10-11:00 VAP 1 - course 46 VAP 3 - course 29 VAP 7 - course 27 VAP 10 - course 22 VAP 17 - course 28 Lecturer 2 - course 16 Lecturer 3 - course 17 Lecturer 10 - course 7 Professor 8 - course 31 Professor 30 - course 49		10:10-11:00 VAP 1 - course 46 VAP 3 - course 29 VAP 7 - course 27 VAP 10 - course 22 VAP 17 - course 28 Lecturer 2 - course 16 Lecturer 3 - course 17 Lecturer 10 - course 7 Professor 8 - course 31 Professor 30 - course 49		10:10-11:00 VAP 1 - course 46 VAP 3 - course 29 VAP 7 - course 27 VAP 10 - course 22 VAP 17 - course 28 Lecturer 2 - course 16 Lecturer 3 - course 17 Lecturer 10 - course 7 Professor 8 - course 31 Professor 30 - course 49
11:15-12:05 VAP 8 - course 33 Lecturer 2 - course 9 Professor 31 - course 55	11:30-12:45 VAP 4 - course 23 Lecturer 4 - course 2 Lecturer 12 - course 4 Professor 19 - course 60 Professor 22 - course 42	11:15-12:05 VAP 8 - course 33 Lecturer 2 - course 9 Professor 31 - course 55	11:30-12:45 VAP 4 - course 23 Lecturer 4 - course 2 Lecturer 12 - course 4 Professor 19 - course 60 Professor 22 - course 42	11:15-12:05 VAP 8 - course 33 Lecturer 2 - course 9 Professor 31 - course 55
12:20-1:10 Lecturer 3 - course 13 Professor 7 - course 40 Professor 17 - course 24 Professor 25 - course 50	1:00-2:15 VAP 15 - course 46 Lecturer 9 - course 17 Professor 10 - course 40 Professor 15 - course 56	12:20-1:10 Lecturer 3 - course 13 Professor 7 - course 40 Professor 17 - course 24 Professor 25 - course 50	1:00-2:15 VAP 15 - course 46 Lecturer 9 - course 17 Professor 10 - course 40 Professor 15 - course 56	12:20-1:10 Lecturer 3 - course 13 Professor 7 - course 40 Professor 17 - course 24 Professor 25 - course 50
1:25-2:15 VAP 1 - course 34 VAP 2 - course 24 VAP 3 - course 29 VAP 6 - course 25 Lecturer 13 - course 8 Lecturer 14 - course 18 Professor 6 - course 57 Professor 32 - course 58 Professor 36 - course 26		1:25-2:15 VAP 1 - course 34 VAP 2 - course 24 VAP 3 - course 29 VAP 6 - course 25 Lecturer 13 - course 8 Lecturer 14 - course 18 Professor 6 - course 57 Professor 32 - course 58 Professor 36 - course 26		1:25-2:15 VAP 1 - course 34 VAP 2 - course 24 VAP 3 - course 29 VAP 6 - course 25 Lecturer 13 - course 8 Lecturer 14 - course 18 Professor 6 - course 57 Professor 32 - course 58 Professor 36 - course 26
2:30-3:45 VAP 6 - course 26 VAP 16 - course 19 VAP 18 - course 28 Lecturer 6 - course 13 Lecturer 7 - course 20 Lecturer 10 - course 13 Professor 5 - course 26 Professor 7 - course 32 Professor 18 - course 29 Professor 23 - course 38 Professor 27 - course 48 Professor 35 - course 59 Professor 36 - course 26	2:30-3:45 VAP 5 - course 35 VAP 12 - course 41 Lecturer 8 - course 17 Lecturer 9 - course 14 Lecturer 11 - course 14 Lecturer 12 - course 4 Professor 3 - course 45 Professor 11 - course 51 Professor 12 - course 27 Professor 33 - course 28	2:30-3:45 VAP 6 - course 26 VAP 16 - course 19 VAP 18 - course 28 Lecturer 6 - course 13 Lecturer 7 - course 20 Lecturer 10 - course 13 Professor 5 - course 26 Professor 7 - course 32 Professor 18 - course 29 Professor 23 - course 38 Professor 27 - course 48 Professor 35 - course 59 Professor 36 - course 26	2:30-3:45 VAP 5 - course 35 VAP 12 - course 41 Lecturer 8 - course 17 Lecturer 9 - course 14 Lecturer 11 - course 14 Lecturer 12 - course 4 Professor 3 - course 45 Professor 11 - course 51 Professor 12 - course 27 Professor 33 - course 28	2:30-3:45
4:00-5:15 VAP 8 - course 46 Lecturer 1 - course 18 Lecturer 13 - course 8 Professor 20 - course 47 Professor 23 - course 39 Professor 24 - course 61	4:00-5:15 Lecturer 4 - course 2 Professor 9 - course 25 Professor 28 - course 52 Professor 29 - course 37	4:00-5:15 VAP 8 - course 46 Lecturer 1 - course 18 Lecturer 13 - course 8 Professor 20 - course 47 Professor 23 - course 39 Professor 24 - course 61	4:00-5:15 Lecturer 4 - course 2 Professor 9 - course 25 Professor 28 - course 52 Professor 29 - course 37	4:00-5:15
5:30-6:45	5:30-6:45	5:30-6:45	5:30-6:45	5:30-6:45

Figure 7: Schedule 2 Pre-Registration Changes



Monday	Tuesday	Wednesday	Thursday	Friday
8:00-8:50 Professor 36 - course 26	**8:30-9:45** VAP 8 - course 17 VAP 14 - course 28 VAP 18 - course 37 Lecturer 1 - course 18 Lecturer 8 - course 17 Lecturer 11 - course 14 Professor 5 - course 41 Professor 8 - course 27 Professor 36 - course 26	**8:00-8:50** Professor 36 - course 26	**8:30-9:45** VAP 8 - course 17 VAP 14 - course 28 VAP 18 - course 37 Lecturer 1 - course 18 Lecturer 8 - course 17 Lecturer 11 - course 14 Professor 5 - course 41 Professor 8 - course 27 Professor 36 - course 26	**8:00-8:50** Professor 36 - course 26
9:05-9:55 Lecturer 3 - course 4 Lecturer 4 - course 18 Lecturer 11 - course 17 Lecturer 14 - course 9 Professor 7 - course 46 Professor 12 - course 11 Professor 13 - course 28 Professor 20 - course 47 Professor 25 - course 31 Professor 29 - course 23	**10:00-11:15** VAP 7 - course 15 VAP 15 - course 46 Lecturer 1 - course 18 Lecturer 9 - course 13 Lecturer 10 - course 17 Lecturer 12 - course 4 Professor 3 - course 45 Professor 17 - course 27 Professor 22 - course 38 Professor 23 - course 26 Professor 38 - course 28	**9:05-9:55** Lecturer 3 - course 4 Lecturer 4 - course 18 Lecturer 11 - course 17 Lecturer 14 - course 9 Professor 7 - course 46 Professor 12 - course 11 Professor 13 - course 28 Professor 20 - course 47 Professor 25 - course 31 Professor 29 - course 23	**10:00-11:15** VAP 7 - course 15 VAP 15 - course 46 Lecturer 1 - course 18 Lecturer 9 - course 13 Lecturer 10 - course 17 Lecturer 12 - course 4 Professor 3 - course 45 Professor 17 - course 27 Professor 22 - course 38 Professor 23 - course 26 Professor 38 - course 28	**9:05-9:55** Lecturer 3 - course 4 Lecturer 4 - course 18 Lecturer 11 - course 17 Lecturer 14 - course 9 Professor 7 - course 46 Professor 12 - course 11 Professor 13 - course 28 Professor 20 - course 47 Professor 25 - course 31 Professor 29 - course 23
10:10-11:00 VAP 9 - course 32 VAP 12 - course 36 VAP 18 - course 16 Lecturer 2 - course 5 Lecturer 3 - course 12 Lecturer 6 - course 4 Professor 4 - course 42 Professor 6 - course 57 Professor 37 - course 34		**10:10-11:00** VAP 9 - course 32 VAP 12 - course 36 VAP 18 - course 16 Lecturer 2 - course 5 Lecturer 3 - course 12 Lecturer 6 - course 4 Professor 4 - course 42 Professor 6 - course 57 Professor 37 - course 34		**10:10-11:00** VAP 9 - course 32 VAP 12 - course 36 VAP 18 - course 16 Lecturer 2 - course 5 Lecturer 3 - course 12 Lecturer 6 - course 4 Professor 4 - course 42 Professor 6 - course 57 Professor 37 - course 34
11:15-12:05 VAP 1 - course 27 VAP 12 - course 40 Lecturer 2 - course 16 Lecturer 3 - course 13 Lecturer 4 - course 2 Lecturer 6 - course 5 Lecturer 7 - course 20 Lecturer 13 - course 8 Professor 25 - course 50 Professor 32 - course 53	**11:30-12:45** VAP 4 - course 24 VAP 5 - course 35 VAP 7 - course 16 VAP 8 - course 17 Lecturer 1 - course 7 Lecturer 10 - course 13 Professor 11 - course 51 Professor 28 - course 52 Professor 30 - course 49 Professor 35 - course 59	**11:15-12:05** VAP 1 - course 27 VAP 12 - course 40 Lecturer 2 - course 16 Lecturer 3 - course 13 Lecturer 4 - course 2 Lecturer 6 - course 5 Lecturer 7 - course 20 Lecturer 13 - course 8 Professor 25 - course 50 Professor 32 - course 53	**11:30-12:45** VAP 4 - course 24 VAP 5 - course 35 VAP 7 - course 16 VAP 8 - course 17 Lecturer 1 - course 7 Lecturer 10 - course 13 Professor 11 - course 51 Professor 28 - course 52 Professor 30 - course 49 Professor 35 - course 59	**11:15-12:05** VAP 1 - course 27 VAP 12 - course 40 Lecturer 2 - course 16 Lecturer 3 - course 13 Lecturer 4 - course 2 Lecturer 6 - course 5 Lecturer 7 - course 20 Lecturer 13 - course 8 Professor 25 - course 50 Professor 32 - course 53
12:20-1:10 VAP 2 - course 46 VAP 9 - course 14 Lecturer 4 - course 2 Lecturer 5 - course 16 Lecturer 7 - course 18 Professor 7 - course 30 Professor 9 - course 25 Professor 15 - course 56 Professor 24 - course 61	**1:00-2:15** VAP 3 - course 29 VAP 15 - course 39 VAP 16 - course 19 VAP 17 - course 24 Lecturer 8 - course 18 Lecturer 10 - course 13 Lecturer 12 - course 4 Lecturer 14 - course 6 Professor 8 - course 26 Professor 27 - course 48 Professor 31 - course 55	**12:20-1:10** VAP 2 - course 46 VAP 9 - course 14 Lecturer 4 - course 2 Lecturer 5 - course 16 Lecturer 7 - course 18 Professor 7 - course 30 Professor 9 - course 25 Professor 15 - course 56 Professor 24 - course 61	**1:00-2:15** VAP 3 - course 29 VAP 15 - course 39 VAP 16 - course 19 VAP 17 - course 24 Lecturer 8 - course 18 Lecturer 10 - course 13 Lecturer 12 - course 4 Lecturer 14 - course 6 Professor 8 - course 26 Professor 27 - course 48 Professor 31 - course 55	**12:20-1:10** VAP 2 - course 46 VAP 9 - course 14 Lecturer 4 - course 2 Lecturer 5 - course 16 Lecturer 7 - course 18 Professor 7 - course 30 Professor 9 - course 25 Professor 15 - course 56 Professor 24 - course 61
1:25-2:15 VAP 1 - course 46 VAP 6 - course 25 VAP 14 - course 22 Lecturer 5 - course 3 Lecturer 11 - course 14 Lecturer 13 - course 8 Professor 10 - course 27 Professor 32 - course 58		**1:25-2:15** VAP 1 - course 46 VAP 6 - course 25 VAP 14 - course 22 Lecturer 5 - course 3 Lecturer 11 - course 14 Lecturer 13 - course 8 Professor 10 - course 27 Professor 32 - course 58		**1:25-2:15** VAP 1 - course 46 VAP 6 - course 25 VAP 14 - course 22 Lecturer 5 - course 3 Lecturer 11 - course 14 Lecturer 13 - course 8 Professor 10 - course 27 Professor 32 - course 58
2:30-3:45 VAP 2 - course 35 VAP 6 - course 26 Lecturer 2 - course 10 Lecturer 6 - course 13 Lecturer 7 - course 17 Lecturer 13 - course 16 Professor 16 - course 40 Professor 33 - course 26 Professor 37 - course 44	**2:30-3:45** VAP 3 - course 29 VAP 13 - course 40 Lecturer 9 - course 18 Lecturer 14 - course 14 Professor 11 - course 54 Professor 17 - course 24 Professor 35 - course 43	**2:30-3:45** VAP 2 - course 35 VAP 6 - course 26 Lecturer 2 - course 10 Lecturer 6 - course 13 Lecturer 7 - course 17 Lecturer 13 - course 16 Professor 16 - course 40 Professor 33 - course 26 Professor 37 - course 44	**2:30-3:45** VAP 3 - course 29 VAP 13 - course 40 Lecturer 9 - course 18 Lecturer 14 - course 14 Professor 11 - course 54 Professor 17 - course 24 Professor 35 - course 43	**2:30-3:45**
4:00-5:15 VAP 10 - course 35 Professor 12 - course 33	**4:00-5:15** VAP 4 - course 26 VAP 13 - course 40 VAP 16 - course 25 Lecturer 8 - course 17 Lecturer 9 - course 18 Lecturer 12 - course 1 Professor 18 - course 29 Professor 19 - course 60 Professor 23 - course 13	**4:00-5:15** VAP 10 - course 35 Professor 12 - course 33	**4:00-5:15** VAP 4 - course 26 VAP 13 - course 40 VAP 16 - course 25 Lecturer 8 - course 17 Lecturer 9 - course 18 Lecturer 12 - course 1 Professor 18 - course 29 Professor 19 - course 60 Professor 23 - course 13	**4:00-5:15**
5:30-6:45	5:30-6:45	5:30-6:45	5:30-6:45	5:30-6:45

Figure 8: Schedule 2 Post-Registration Changes