SCHEDULING CLUE MATH TUTORS FOR AUTUMN QUARTERS

LIAM CHU, DIANNA E*, TYLER FRANCK*, SARAH MATHISON*, PUJAN PATEL

Abstract. The Center for Learning and Undergraduate Enrichment (CLUE) is an academic support program at the University of Washington that provides drop-in tutoring for a wide range of subjects. Historically, the demand for math tutoring has vastly exceeded that of other subjects, and recently, the program has struggled to meet this demand. This is particularly true for autumn quarter as students, refreshed from summer break, are especially energized and motivated to succeed. Unfortunately, the inability to meet this demand means that many students are often turned away without receiving any help at all. In this paper, we hope to bridge the gap between supply and demand by developing software to more intelligently schedule tutors according to the trends of past autumn quarters. More specifically, we develop a non-linear programming model to find the weekly tutoring schedule that best aligns with the predicted daily demand whilst adhering to the personal and institutional constraints put forth by tutors and program management.

1. Introduction and Overview

1.1. Background.

The Center for Learning and Undergraduate Enrichment (CLUE) is an academic support program at the University of Washington that provides drop-in math tutoring from 7pm to 11pm on Sunday through Thursday nights. Students can get tutoring either in-person at Mary Gates Hall or virtually via Zoom. Regardless of the format, the process is the same: upon arrival, students enter a waiting queue, and tutors serve students following the order of that queue. While sessions typically take between 20 and 30 minutes, they can be shorter or longer depending on the complexity/nature of the problem or the attendance for the night. As such, a single tutor may help between 8 to 12 students a night.

In the past few years the demand for math tutoring in the autumn has risen dramatically, with some nights seeing upwards of 100 students, and CLUE's ability to satisfy this demand has lagged behind. The most recent autumn quarter, for example, had roughly 26% of attending students leave with out receiving any help at all [4]. This is problematic not only because these students' needs remain unmet but also because autumn quarter is most students' first contact with CLUE: a negative first experience may give them the impression that it is not worth coming back again. In this way, CLUE's ability to accommodate the high demand during autumn quarter is fundamental to their mission of providing academic support to as many students as possible, both in that present autumn quarter and in the subsequent quarters as well.

The natural response to an overwhelming demand for math tutoring is to hire more math tutors, and that's exactly what program management did. As of spring 2024, we currently have 18 math tutors working at CLUE. While an increase in staffing *should* address the issue, this is only true if we are allocating our resources effectively. Currently, tutor work schedules are manually arranged by the lead tutor(s), a tedious task that requires them to accommodate each individual's availability and preferences while also achieving a balanced coverage of the week. In this project, we seek to develop scalable, data-driven software to automate this process, both for the convenience, flexibility, and efficiency it will give those choosing the schedules and also for the opportunity to leverage the computational power of computers to generate a mathematically optimal schedule.

E-mail address: `liamchu@uw.edu, de33@uw.edu, tfranck@uw.edu, smathi2@uw.edu, pujanp@uw.edu`.

Date: June 5, 2024.

* As you read the paper, you might encounter us using the word "we" in a way that would suggest we were part of CLUE because some of the authors *are* a part of CLUE. In particular, Dianna, Tyler, and Sarah work as math tutors.

1.2. Goals.

The goal of our project is to solve a reoccurring problem: we want to develop a tool that finds the weekly tutoring schedule (for a given set of tutors) that maximizes the number of students helped subject to (1) the personal and institutional preferences/constraints put forth by tutors and program management and (2) the projected daily demand based on past autumn quarters. We emphasize that the goal is to find a reusable solution, not a one-time answer. We are not trying to find the best schedule for a *particular* autumn quarter (e.g., autumn 2024), but a model that can be used to find the best schedule for *any* given autumn quarter. This means that the model needs to be configurable rather than hard-coded.

2. Literature Review and Theoretical Background

In this section, we review existing literature on scheduling models, particularly focusing on Integer Programming (IP) and Non-linear Programming (NLP), to establish a theoretical basis for the proposed scheduling model.

2.1. Literature Review.

Integer Programming and Non-linear Programming are employed extensively in complex scheduling scenarios across various domains to manage multiple and competing criteria effectively. For instance, Cocchi et al. (2018) [5] illustrate MIP's utility in the sports sector by optimizing the Italian National Volleyball Tournament schedule–addressing challenges like minimizing travel and balancing games. This parallels challenges we face in scheduling tutors, like balancing the number of tutors on each night and minimizing the number of tutor scheduling preferences that are not satisfied. Similarly, Butar-Butar et al. (2023) [2] demonstrate the application of mixed integer non-linear programming in optimizing energy usage in production schedules, a methodology that can optimize educational processes where constraints such as tutor availability and tutor work preferences are non-linearly related. These studies underscore the practical integration of IP and NLP in addressing real-world problems.

2.2. Integer Programming.

In our model we use a 0-1 Integer Program to efficiently schedule CLUE Math tutors. A 0-1 IP is a optimization problem where the decision variables can only take on the values 0 or 1 [7]. Since a tutor is only ever scheduled or not scheduled, our problem is intrinsically binary. This allows us to easily translate our problem into a 0-1 program. In doing so we simplify the complex entanglement of interdependent decisions that are made in tutor scheduling into a series of simple yes-no decisions. There are various techniques that are used to solve 0-1 IP. We employed Gurobi Optimizer to solve our 0-1 IP. Gurobi Optimizer uses many techniques to efficiently solve integer programs. One common method Gurobi employs is Branch and Bound [1]. We recommend reading *Operations Research* Chapter 9 by Wayne L. Winston for a more indepth look at 0-1 programs, including the various uses for 0-1 programs and the techniques used solve these types of optimization problems [7].

2.3. Non-linear Programming (NLP).

We utilize an Integer Quadratic Program (IQP) in our model. This is a type of integer optimization problem in which we optimize a quadratic function with respect to linear constraints [1]. We use a quadratic objective function to create a homogenized distribution of "error" (in our case error means over or under staffing). This is similar to the balance metric used in "The Missouri Lottery..." by Wooseung Jan et al. to measure the squared difference between the length of individual routes and the average route length [6]. It is important to note that there are linear components of our objective function that we optimize, such as the number of shifts and adherence to tutor preferences in addition to our quadratic term.

3. Model Description and Formulation Process

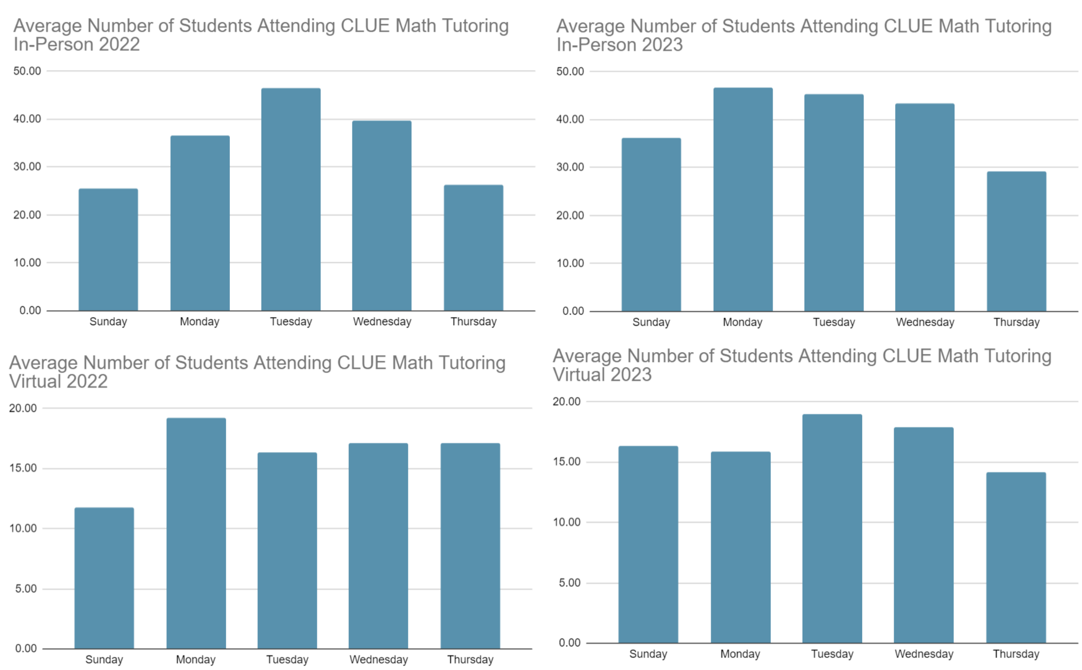
3.1. **Simplifications and Limitations.**

When tackling complex real-world problems, it is often impossible to make a mathematical model that both perfectly captures the problem *and* is practical. Indeed, constructing mathematical models is often an act of balancing simplicity and detail: the model needs to be simple enough to use and solve but detailed enough so that the solution is meaningful and useful. It is for this reason that translating real-world problems into mathematical problems almost always requires making some simplifying assumptions along the way. Here, we describe the assumptions we made while designing our model.

(1) We assume that the trends of past autumn quarters will persist into future autumn quarters.

One of the goals of this project is to use the data of past quarters to inform the scheduling of future quarters. This implicitly assumes that the two aren't completely unrelated, that knowing the past will actually tell you *something* about the future. While this assumption makes sense, it is not at all guaranteed to be true. For example, if assignment due dates shift from Friday one quarter to Wednesday the next, then we'd likely see attendance shift earlier in the week to match this (because students tend to do their assignments right before the deadline). In Figure 1, we see there do appear to be *some* trends across autumn quarters: for instance, Tuesday is consistently one of CLUE's busiest days while Thursday and Sunday are CLUE's slowest. That being said, two quarters' worth of data isn't that convincing. With more data, we could hope to gather more evidence and determine whether this assumption is truly justified, however, we just can't say with the data we have.

To address this uncertainty, we designed the model so that this past quarter data is incorporated via an *input variable* that tells the model how it should aim to distribute shifts across the week days (e.g., schedule 20% of shifts on Tuesday in-person). We use the past trends in student attendance to create a recommended value for this input variable, but the user can simply choose to use a different value if they want. In this way, the model can still be used even if the past trends are no longer applicable.



[3] [4]

FIGURE 1. Average student attendance in 2022 and 2023.

(2) We assume that scheduling tutor shifts to match the average daily demand over the week will help the most students.

The fundamental idea behind this assumption is quite simple: if we expect twice as many *students* to attend day_1 than day_2, then we should schedule twice as many *shifts* on day_1 than on day_2. Of course, reality is not so simple, and this assumption actually relies on two smaller assumptions. For one, we must assume that the average demand on a week day (over the entire quarter) will accurately reflect the demand on that day throughout the quarter. It would be bad if, for example, a week day had a high average attendance because of only a few days that were extremely busy. Secondly, we must also assume that each tutor is equally productive. If certain tutors are significantly more productive than others, then merely considering the *number* of scheduled shifts would be insufficient; we would also need to account for *who* was working. CLUE currently doesn't track individual performance and it wouldn't have any data on new tutors even if it did, and so we will just assume that all tutors are indistinguishable when it comes to productivity.

(3) Our model is not designed for the first week, dead week (the week before finals week), or finals week.

CLUE has a special tutoring schedule for these weeks because demand is abnormal. Since our model is intended to construct a schedule for the quarter overall, it does not make sense to consider these outliers.

(4) Tutors can only work full shifts (4 hours).

When filling out the tutor scheduling form, only full shifts are available. Half shifts are uncommon and are typically only implemented if CLUE is very short staffed. Full shifts are preferred and typical, so we assume all tutors are working only full shifts.

3.2. Model Overview.

Having described the assumptions that underlie the design of our model, it's time to talk about the model itself. But, before diving into all the mathematical details, it will be helpful to get a conceptual overview of all the different pieces of the model and what they do. Our model consists of three components: the variables, constraints, and objective function. The variables encapsulate both the user's input to the model (e.g., the number of tutors and their availability) as well as the model's output (e.g., schedule Alice to work in-person on Monday). The constraints use the variables to encode which schedules are allowed and which are not. For example, there is a constraint that prevents a tutor from being scheduled to work both in-person and virtually on the same day. Lastly, the objective function evaluates the "quality" of a schedule by assigning it a numerical value, which the model uses to choose the best schedule among multiple valid options. For example, we might prefer schedules with a higher number of shifts scheduled, and so we would want our objective function to give a higher value to such schedules so that our model will prefer such schedules as well. In summary, the variables encode the problem given to our model and answer it gives back, the constraints encode which answers are allowed, and the objective function encodes which answers are better than others. All of these elements work together to allow the model to find the best schedule.

3.3. Variables.

Input Variables: These are the arguments *given* to our model by the user

- $T = \{1, 2, \ldots, N_T\}$: the set of tutors
- $D = \{1, 2, \ldots, N_D\}$: the set of working days
- ℓ_p: the minimum number of tutors needed in-person each day
- ℓ_v: the minimum number of tutors needed virtually each day
- s: the minimum number of shifts a tutor needs to work per week

- s^p[1]: the minimum number of in-person shifts a tutor needs to work per week
- S_t: the maximum number of shifts tutor t is willing to work per week
- $a_{td} \in \{0, 0.5, 1\}$: the availability/preference of tutor t to work day d
  - 0 means unavailable (i.e., they cannot work that day)
  - 0.5 means not preferred (i.e., they *can* work that day but would rather not)
  - 1 means preferred
- $m_t \in \{-1, 0, 1\}$: the preferred modality of tutor t
  - -1 means that they would prefer to work virtually
  - 1 means that they would prefer to work in-person
  - 0 means that they have no preference
- f_d^p[2]: the target fraction of total shifts to occur on day d in-person
- f_d^v[3]: the target fraction of total shifts to occur on day d virtually
- w_1, w_2, w_3, w_4[4]: weights for the components of the objective function

Decision Variables: These are the variables that our model *chooses* to encode an output schedule

- $x_{td}^p \in \{0, 1\}$: indicates whether tutor t is scheduled to work in-person on day d
- $x_{td}^v \in \{0, 1\}$: indicates whether tutor t is scheduled to work virtually on day d

Derived Variables: These are the variables derived from the other variables in order to make the following constraints and objective function more interpretable

- $x_{td} = x_{td}^p + x_{td}^v \in \{0, 1\}$: indicates whether tutor t is scheduled to work day d (either in-person or virtually)
- $X = \sum_{t,d} x_{td}$: the total number of shifts scheduled per week
- $X_{.d}^p = \sum_t x_{td}^p$: the number of in-person shifts scheduled for day d
- $X_{.d}^v = \sum_t x_{td}^v$: the number of virtual shifts scheduled for day d
- $X_{t.} = \sum_d x_{td}$: the number of shifts scheduled for tutor t
- $X_{t.}^p = \sum_d x_{td}^p$: the number of in-person shifts scheduled for tutor t per week
- $X_{t.}^v = \sum_d x_{td}^v$: the number of virtual shifts scheduled for tutor t per week

3.4. Constraints.

- $x_{td} = x_{td}^p + x_{td}^v \leq 1$: a tutor cannot be scheduled for both an in-person and virtual shift on the same day
- $x_{td} = 0$ whenever $a_{td} = 0$: a tutor cannot be scheduled on a day they're unavailable
- $s \leq X_{t.} \leq S_t$: each tutor must be scheduled for at least s shifts and no more than the maximum number of shifts they are willing to work
- $X_{t.}^p \geq s^p$: each tutor must be scheduled for at least s^p in-person shift(s)
- $X_{.d}^p \geq \ell_p$: each day must have at least ℓ_p tutors scheduled to work in-person
- $X_{.d}^v \geq \ell_v$: each day must have at least ℓ_v tutors scheduled to work virtually

3.5. Objective Function.

[1] You may notice that there is an s^p variable but no s^v variable. Currently, CLUE's contract requires that every tutor works at least 1 shift in-person, but there is no corresponding requirement for virtual shifts. So, it simply doesn't make sense to include such a variable, though the model can be easily tweaked if the need arises in the future.

[2] We recommend using $(f_{\text{Sun}}^p, f_{\text{Mon}}^p, f_{\text{Tue}}^p, f_{\text{Wed}}^p, f_{\text{Thu}}^p) = (0.09, 0.18, 0.175, 0.155, 0.1)$. For each quarter in our dataset, we count the number of students who attended CLUE on day d in-person and divide it by the total number of students who attended CLUE that quarter (ignoring the first week, dead week, and finals week). The recommended value f_d^p is the average of these fractions.

[3] We recommend using $(f_{\text{Sun}}^v, f_{\text{Mon}}^v, f_{\text{Tue}}^v, f_{\text{Wed}}^v, f_{\text{Thu}}^v) = (0.04, 0.075, 0.075, 0.065, 0.045)$.

[4] Based on our testing of the model, we recommend using $(w_1, w_2, w_3, w_4) = (3, 2, 1, 1)$.

$$\text{SHIFTS} = X \tag{1}$$

$$\text{ALIGNMENT} = -\frac{1}{N_T}\sum_{d}\left[\left(f_d^p X - X_{.d}^p\right)^2 + \left(f_d^v X - X_{.d}^v\right)^2\right] \tag{2}$$

$$\text{DAY\_PREF} = -3X + 4\sum_{t,d} a_{td}x_{td} \in [-X, X] \tag{3}$$

$$\text{MODE\_PREF} = \sum_{t,d}\left(x_{td}^p - x_{td}^v\right) m_t \in [-X, X] \tag{4}$$

$$\text{Objective Function} = w_1\text{SHIFTS} + w_2\text{ALIGNMENT} + w_3\text{DAY\_PREF} + w_4\text{MODE\_PREF} \tag{5}$$

Explanation of components of the objective function:

(1) SHIFTS := total number of shifts scheduled per week

(2) ALIGNMENT := measure of how proportional the distribution of tutors in a schedule is to the target fractions.

This term increases when the distribution of tutors in a schedule is more proportional to the target fractions. With well chosen target fractions, the ALIGNMENT term increases when we schedule tutors according to the relative demand for CLUE Math tutoring on each day of the week.

We can think of $f_d^p X - X_{.d}^p$ and $f_d^v X - X_{.d}^v$ as the error between the target distribution of tutors and the actual distribution of tutors in the schedule. We use quadratic terms because we want schedules with homogenized error to be preferred (have a higher \ less negative value) [6]. For example, a schedule where many days have small errors is better than a schedule where one day has a huge error and the rest are perfect. This is because we can not compensate for under staffing one day by over staffing the another day.

(3) DAY_PREF := score that increases if a higher portion of the tutors are scheduled on days they prefer.

If all tutors are scheduled on preferred days then the availability score of every tutor on each day they work is 1. Then $\sum_{t,d} a_{td}x_{td} = \sum_{t,d} x_{td} = X$. Then $-3X + 4\sum_{t,d} a_{td}x_{td} = X$.

If all tutors are scheduled on not preferred days then the availability score of every tutor on each day they work is 0.5. Then $\sum_{t,d} a_{td}x_{td} = 0.5\sum_{t,d} x_{td} = 0.5X$. Then $-3X + 4\sum_{t,d} a_{td}x_{td} = -X$.

(4) MODE_PREF := score that increases if a higher portion of the tutors are scheduled in the mode (in-person or virtual) they prefer.

If all tutors are scheduled only virtually $\sum_{t,d}\left(x_{td}^p - x_{td}^v\right) = \sum_{t,d}\left(-x_{td}^v\right) = -X$. Then if all tutor prefer virtual $\sum_{t,d}\left(x_{td}^p - x_{td}^v\right) m_t = -X * -1 = X$ and if all tutor prefer in-person $\sum_{t,d}\left(x_{td}^p - x_{td}^v\right) m_t = -X * 1 = -X$.

If all tutors are scheduled only in-person $\sum_{t,d}\left(x_{td}^p - x_{td}^v\right) = \sum_{t,d}\left(x_{td}^p\right) = X$. Then if all tutor prefer virtual $\sum_{t,d}\left(x_{td}^p - x_{td}^v\right) m_t = X * -1 = -X$ and if all tutor prefer in-person $\sum_{t,d}\left(x_{td}^p - x_{td}^v\right) m_t = X * 1 = X$.

(5) Weights w_1, w_2, w_3, w_4: While ultimately these weights are input values that are chosen by the user, we chose the default weights $w_1 = 3$, $w_2 = 2$, $w_3 = w_4 = 1$, based on the priority of each term. Our first priority is to schedule as many shifts as possible so w_1 has the highest weight. Our second priority is scheduling these shifts correspond to demand on each day of the week so w_2 has the second highest weight. Our third priority is satisfying tutor preferences so w_3 and w_4 have the lowest weight.

4. Example Problem Formulation

Here, we describe a brief example of how such a problem might be formulated solved.

Suppose we have three math tutors. We analyze the number of students attending CLUE Math Tutoring on each day of the week in past autumn quarters. Given the distribution of demand across days of the week, we aim to find how many tutors we need such that the number of students helped is maximized, while also considering the constraints of tutor availability and preference.

Assume, for a simple example, that we are only planning on staffing Tuesday, Thursday, and Sunday and there are only in-person shifts (so no mode preference). Tutor 1 is willing to work two days, Tutor 2 is willing two days, and Tutor 3 is willing to work all three days. From past data, we find that 30% of students show up on Tuesday, 40% show up on Thursday, and 30% of students show up on Sunday. Assume we receive the following tutor availabilities:

	T1	**T2**	**T3**
Tue	preferred	preferred	not preferred
Thu	not available	preferred	preferred
Sun	preferred	not available	not preferred

Since two tutors must be scheduled every day and we cannot schedule tutors on days they are not available. Then the set of possible feasible schedules are:

**The days each tutor is scheduled are marked by X's*

Schedule 1:

	T1	**T2**	**T3**
Tue	X	X	X
Thu		X	X
Sun	X		X

Schedule 2:

	T1	**T2**	**T3**
Tue	X	X	
Thu		X	X
Sun	X		X

Schedule 3:

	T1	**T2**	**T3**
Tue	X		X
Thu		X	X
Sun	X		X

Schedule 4:

	T1	**T2**	**T3**
Tue		X	X
Thu		X	X
Sun	X		X

We can see Schedule 2 is a better schedule than Schedules 3 and 4 as the only difference between them is that Schedule 2 has two tutors that prefer to work on Tuesday while Schedules 3 and 4 have one tutor that prefers to work and one tutor that does not prefer to work on Tuesday. The advantage of Schedule 2 is represented in our objective function by DAY_PREF term. Then we will necessarily choose Schedule 2 over Schedules 3 and 4.

Looking at Schedules 1 and 2 we can see that the only difference between the schedules is that Schedule 1 places one more tutor on Tuesday than Schedule 2. Then, since we wish to maximize the number of student helped each day, Schedule 1 is the better schedule. Thus, our algorithm should output Schedule

1 as the optimal schedule. To see that our algorithm outputs the desired solution consider our objective function value for Schedule 1 and 2.

**Note that there is no DAY_MODE and ALIGNMENT only contains the in-person sum since we are only scheduling in-person shifts.*

SHIFTS=X
ALIGNMENT=$-\frac{1}{n}\sum_d \left[\left(f_d^p X - X_{.d}^p\right)^2\right]$
DAY_PREF=$-3X + 4\sum_{t,d} a_{td}x_{td} \in [-X, X]$
Objective Function $= w_1$SHIFTS $+ w_2$ALIGNMENT $+ w_3$DAY_PREF

The set of days is $d \in [1, 2, 3]$ (where 1=Tuesday, 2=Thursday, 3=Sunday). Our target fractions, based on the daily demands for CLUE Math Tutoring given above are, $f_1^p = 0.30$, $f_2^p = 0.40$, $f_3^p = 0.30$. The set of tutors is $t \in [1, 2, 3]$. Finally the tutor availabilities given in the table above can be translated to $a_{11} = 1$, $a_{12} = 0$, $a_{13} = 1$, $a_{21} = 1$, $a_{22} = 1$, $a_{23} = 0$, $a_{31} = 0.5$, $a_{32} = 1$, $a_{33} = 0.5$.

We now evaluate the objective function value of Schedule 1. Schedule 1 contains a total of 7 shifts, 3 shifts on Tuesday, 2 shifts on Thursday, and 2 shifts on Sunday. Plugging these values in we find:
SHIFTS= 7,
ALIGNMENT $= -\frac{1}{3}[(7 * 0.30 - 3)^2 + (7 * 0.40 - 2)^2 + (7 * 0.30 - 2)^2] = -0.487$,
DAY_PREF $= -3 * 7 + 4[(1 + 1 + 0.5)\ +\ (1 + 1)\ +\ (1 + 0.5)] = 3$.
This gives us an objective function value of:
Objective Function= 3*SHIFTS + 2*ALIGNMENT+DAY_PREF= 23.026.

We now evaluate the objective function value of Schedule 2. Schedule 2 contains a total of 6 shifts, 2 shifts on Tuesday, 2 shifts on Thursday, and 2 shifts on Sunday. Plugging these values in we find:
SHIFTS= 6,
ALIGNMENT $= -\frac{1}{3}[(6 * 0.30 - 2)^2 + (6 * 0.40 - 2)^2 + (6 * 0.30 - 2)^2] = -0.08$,
DAY_PREF $= -3 * 6 + 4[(1 + 1)\ +\ (1 + 1)\ +\ (1 + 0.5)] = 4$.
Note that the ALIGNMENT value is greater than the ALIGNMENT value for Schedule 1. This is because the distribution of tutors across days of the week in Schedule 2 is more proportionate to the target fractions. Similarly notice that DAY_PREF is higher, as all the tutors were scheduled on days they prefer. In this way, we can see each term behaves as we would expect.
This gives us an objective function value of:
Objective Function=3*SHIFTS+2*ALIGNMENT+DAY_PREF = 21.84.

Then since the objective function value of Schedule 1 is greater than the objective function value of Schedule 2, we know Schedule 1 is slightly better than Schedule 2. From this we can see that our model correctly identified the best schedule, based on our priorities.

It is important to note that with different weights, Schedule 1 may not be the optimal solution.

5. Algorithm Implementation

To solve our non-linear binary optimization problem, we used Gurobi Optimizer. Gurobi 11.0.0 supports solving non-linear integer programs over convex regions using spatial branch-and-bound and outer approximation to find globally optimal solution.

5.1. Inputs and Constraints.

Constructing our model in Gurobi is a straightforward process, thanks its simple method for specifying an objective function with associated variables and constraints. Input variables are tabled below with the associated mathematical symbol from sections 3.3 (model variables), as well as their Python data structure.

Input	Math Symbol	Data Structure
Tutor Names	T	List
Working Days	D	List
Min. In-person Shifts	ℓ_p	Numeric
Min. Virtual Shifts	ℓ_v	Numeric
Min. Weekly Shifts per Tutor	s	Numeric
Min. In-person Shifts per Tutor	s_p	Numeric
Max Shifts per Tutor	S_t	Dictionary
Tutor Daily Availability	a_{td}	Dictionary
Tutor Modality Preference	m_t	Dictionary
Target Fraction In-person	f_d^p	Dictionary
Target Fraction Virtual	f_d^v	Dictionary

Constraints utilize the input variables from above and are encoded according to the specification of Gurobi's API.

5.2. **Objective Function.**
After input variables and constraints are assigned, we specify the objective function, and solve the optimization problem using Gurobi's `model.optimize()` function. We note that our CLUE contact dictated that if Gurobi is unable to produce an optimal result, we should simply report that there is no feasible schedule using the given input data. However, we are told that the possibility of this happening is nearly impossible, and not to place much emphasis on finding a solution for this scenario.

6. Results

6.1. **Verification and Testing.**
In this section we demonstrate the efficacy our model by analyzing the output of our program on a realistic test case. Unfortunately, due to privacy concerns we cannot use real tutor availability forms from past autumn quarters. Then to test our model we constructed a realistic set of tutor availabilities.

Some common trends in tutor availability that our community partner informed us of and that we have noticed as CLUE Math Tutors are:

- The large majority of tutors want to work two shifts a week (no less as this breaks the contract).
- Tutors almost never want to work more than three shifts a week.
- There are almost always enough tutors willing to work some day of the week to meet the minimum scheduling requirements (2 in-person and 2 virtual).

Using this information we constructed to following tutor availabilities:

Name	Mode Preference	Max # Shifts (Per Week)	Sun	Mon	Tues	Weds	Thurs
Arnold	In-person	2	Prefered	Prefered	Not Prefered	Unavaible	Not Prefered
Bob	In-person	2	Prefered	Not Prefered	Unavaible	Prefered	Not Prefered
Charles	In-person	2	Not Prefered	Prefered	Prefered	Unavaible	Unavaible
Diana	In-person	3	Not Prefered	Unavaible	Not Prefered	Prefered	Prefered
Evelyn	In-person	2	Unavaible	Prefered	Not Prefered	Prefered	Unavaible
Frederick	Virtual	3	Not Prefered	Not Prefered	Prefered	Prefered	Unavaible
Geovanni	Virtual	2	Not Prefered	Not Prefered	Not Prefered	Not Prefered	Not Prefered
Henrietta	Virtual	2	Not Prefered	Not Prefered	Prefered	Unavaible	Prefered
Isaac	Virtual	2	Prefered	Unavaible	Prefered	Unavaible	Prefered
Juliet	Virtual	2	Prefered	Prefered	Not Prefered	Unavaible	Not Prefered
Karen	No Preference	2	Unavaible	Prefered	Prefered	Unavaible	Unavaible
Lissandra	No Preference	2	Prefered	Not Prefered	Prefered	Not Prefered	Prefered
Maxine	No Preference	2	Not Prefered	Not Prefered	Unavaible	Not Prefered	Prefered
Norman F.R.	No Preference	2	Unavaible	Unavaible	Prefered	Prefered	Not Prefered
Ollie	No Preference	2	Prefered	Not Prefered	Unavaible	Prefered	Not Prefered

We choose the target fractions:

**Where 0 = Sunday, 1 = Monday, 2 = Tuesday, 3 = Wednesday, 4= Thursday*

$f_0^p = 0.05 \qquad f_0^v = 0.1$
$f_1^p = 0.1 \qquad f_1^v = 0.05$
$f_2^p = 0.2 \qquad f_2^v = 0.05$
$f_3^p = 0.2 \qquad f_3^v = 0.05$
$f_4^p = 0.1 \qquad f_4^v = 0.1$

**Remember that target fractions represent the expected distribution of the demand for CLUE Math Tutoring throughout the week. The target fraction we chose have a clear ranking (i.e. 0.05 × 2 = 0.1 and 0.1 × 2 = 0.2) which will make it easier to verify the validity of the model output.*

We set all other inputs to their default values (according to CLUE policy/practice).
Our model outputs the schedule below:

P = scheduled for in-person
V = scheduled for virtual
X = not scheduled

	Sun	Mon	Tue	Wed	Thu	#P	#V
Arnold	P	P	X	X	X	2	0
Bob	P	X	X	P	X	2	0
Charles	X	P	P	X	X	2	0
Diana	X	X	P	P	P	3	0
Evelyn	X	P	X	P	X	2	0
Frederick	V	X	P	V	X	1	2
Geovanni	X	V	X	P	X	1	1
Henrietta	X	X	V	X	P	1	1
Isaac	X	X	V	X	P	1	1
Juliet	V	P	X	X	X	1	1
Karen	X	V	P	X	X	1	1
Lissandra	X	X	P	X	V	1	1
Maxine	X	X	X	P	V	1	1
Norman F.R.	X	X	P	V	X	1	1
Ollie	V	X	X	P	X	1	1
#P	2	4	6	6	3		
#V	3	2	2	2	2		

Before we evaluate the model output lets consider what makes a good schedule.

A good schedule should have the following properties (in order of importance):
(1) The schedule is feasible \ can realistically be implemented by CLUE.
(2) As many shifts are scheduled as possible.
(3) The shifts are scheduled ”smartly” (days with the highest demand for tutors are prioritized).
(4) Tutors preferences are respected.

First it can easily be checked that all of our hard constraints are satisfied.

✓ No tutor works on days they are unavailable.
✓ No tutor is scheduled for multiple shifts on the same day.
✓ Every tutor works at least two shifts a week and at least one in-person shift a week.

Then our model has found a feasible schedule that can realistically be implemented by CLUE.

We will now evaluate how qualitatively "good" this schedule is under our assumptions.

First observe that the schedule above contains a total 32 shifts. Since there are 15 tutors, 13 of which want to work 2 shifts a week and 2 of which want to work 3 shifts a week, we can schedule a maximum of 32 shifts a week. Then our model schedules the maximum number of shifts possible. Since more tutors can help more students, scheduling the maximum number of tutors a week will allow CLUE to help the most students. Then in this regard, the schedule output by our model is what we would hope for.

Now lets check that the tutors are distributed smartly across the days of the week.

Based on our target fractions, the least tutors are need in-person Sunday, virtual Monday, virtual Tuesday, and virtual Wednesday. From the schedule above we can see that 2 tutors are scheduled in the specified mode on each of these days.

Ideally we will have twice as many tutors Monday in-person, Thursday in-person, Sunday virtual, and Thursday virtual as the target fraction suggest these days have twice the demand for tutoring as the slowest days. From the schedule above we can see the model scheduled 4 tutors Monday in-person, 3 tutors Thursday in-person, 3 tutors Sunday virtual, and 2 tutors Thursday virtual. While 4 tutors on each of these day in the corresponding modes would be ideal, since our schedule contains the maximum number of shifts possible, scheduling more tutors on these shifts means taking tutors from other shifts.

The most tutors are needed on Tuesday in-person, and Wednesday in-person. Based on our target fractions four times as many tutors are needed on these days than our slowest days. Then ideally 8 tutors would be scheduled on Tuesday and Wednesday in-person. We can see from the schedule above that 6 tutors are scheduled in-person Tuesday and Wednesday. Since our schedule contains the maximum number of shifts possible scheduling 8 shifts on these days would force us to take tutors from other shifts. There are no shifts with "extra" tutors so the schedule output by the model has the best possible distribution of tutors while scheduling the maximum number of tutors possible.

We will now check that the model satisfies preference whenever possible.

By comparing the schedule output by our model to the tutor availability table we can see that the only tutors that are scheduled on days they do not prefer are Diana, Frederick, Geovanni, and Maxine. We can see that each of these tutors are scheduled for all of their preferred shifts in addition to some not preferred shift(s). Then to reach the maximum number of shifts each of these tutors are willing to work, these tutors had to be scheduled on some day(s) they do not prefer to work. Since we prioritize maximizing the number of shifts, scheduling these tutors on days they do not prefer to work is justified.

Similarly we can see that any tutor that prefers virtual shifts is only scheduled for one in-person shift (as required by the contract). All the tutors that prefer in-person shifts are only scheduled for in-person shifts.

Based on our criteria above for a "good" schedule and our analysis, we can see that the schedule output by our model is indeed a good feasible schedule.

6.2. **Improvements.**

In this section we discuss elements of our model that could potentially be improved in the future. This includes default input values, the objective function, and user interface.

• Currently the values of the target fractions, f_d^p and f_d^v, default to the average fraction of students attending CLUE Math, either virtual or in-person, on the corresponding day of the week out of the total number of students attending CLUE Math in autumn quarter. Since we only have attendance data from two past autumn quarters it is difficult to determine whether these fractions accurately predicts the distribution of demand for CLUE Math Tutoring. With more data we could likely find default values for the target fractions that more precisely match the actual distribution of demand for CLUE.

With data on more past autumn quarter perhaps there will be clear trend in the weekly distribution of demand for CLUE Math Tutoring. If not, we may be able to find better target fractions by considering the homework deadlines in the calculus series (as a large portion of students ask calculus questions). As of now CLUE does not have access to this information, but if this changes this information could be a good predictor of the weekly distribution of demand for CLUE Math Tutoring.

• There is potential for improvement in our objective function. Ideally each term in our objective function (SHIFTS, ALIGNMENT, DAY_PREF, MODE_PREF) would be completely independent of each other. As the objective function is written now the SHIFTS and ALIGNMENT terms are theoretically interdependent. However, in practice X (the number of shifts) is bounded between $5n$ (5 days in the work week) and $2n$ (minimum shifts required to work). Then dividing by n is essentially dividing by X. Because of this the interdependence should not be an issue in practice. This being said, it would be preferable if all terms were completely independent. This is a potential future improvement to the objective function.

• Since our model is meant to be tool for CLUE to use repeatedly, having an intuitive user interface is very important. As of now, all the tutor availability information has to be input by hand. While this is doable, writing a function that would extract the availability data directly from the google forms tutors fill out would streamline this process.

• Although it is unlikely that there is no feasible schedule, in the case that a schedule is infeasible it would be helpful for the program to output what constraints caused the schedule to be infeasible. For example, did a tutor say they are not available every day of the week. The model could then drop the constraints that make the model infeasible in order to find a solution. Implementing this is somewhat complicated since there are some constraints, such as tutor availability, that we could potentially drop to make a feasible schedule while some constraints, such as the minimum number if shifts a tutor in contractually obligated to work cannot be dropped.

7. Summary and Conclusions

We developed a model to optimize the scheduling of CLUE Math tutors for autumn quarters. Using Gurobi Optimizer, we formulated and solved the non-linear binary optimization problem, considering various constraints and objectives.

The developed model effectively schedules CLUE Math tutors, maximizing the number of shifts while aligning with the weekly distribution of demand for tutoring and considering tutor preferences. By integrating these elements into the scheduling process, the model ensures a balanced and efficient allocation of tutoring resources, which can help CLUE provide optimal support to students.

Through this process we learned about the challenges one encounters when translating a complex and multifaceted real world problem into a mathematical model. One complication we encountered was balancing competing objectives. In our model we maximize the number of shifts, align tutor schedules with demand, and respect tutor preferences. Figuring out how to translate each of these problems into a mathematical functions forced us to consider how to quantify qualities that are not innately quantitative. Even after we quantified these terms, we had to think about how to prioritize the objectives. This was especially challenging in the sense that there is not one steadfast answer. We were forced to really consider what our goal is what our community partner wants in order to decide how our objectives should be prioritized. Through this we learned about how to justify the decisions we made and address our limitation.

Appendix

Full Code: https://github.com/pujan-p/CLUE-MATH-SCHEDULING/tree/main

Acknowledgements

We extend our heartfelt gratitude to CLUE for their support and partnership in this project. Their dedication to fostering academic excellence and their commitment to student success has been inspirational and we are so grateful to have had the opportunity to contribute to this mission. We would like to give a special thanks to Ryan Burt for his help on this project. Ryan's guidance was invaluable and his contagious passion for CLUE's mission quickly became a driving force behind the project.

References

[1] Mixed-integer Programming (MIP) – A Primer on the Basics. `https://www.gurobi.com/resources/mixed-integer-programming-mip-a-primer-on-the-basics/`, 2024. Accessed: May 24, 2024.

[2] T. K. N. Butar-Butar, H. Mawengkang, and S. Sitorus. *Mixed Integer Nonlinear Programming Model for Sustainable Production Planning and Scheduling Based on Energy Consumption*. AIP Conference Proceedings, 2023.

[3] CLUE. Clue Nightly Number for 2022-2023 Academic Year. `https://docs.google.com/spreadsheets/d/1oX0-kQ-mRI_aRUaUpRc23DLdAh_CNgjJq5yrS2E1ots/edit#gid=1393586623`, 2022.

[4] CLUE. Clue Nightly Number for 2023-2024 Academic Year. `https://docs.google.com/spreadsheets/d/1beMpqynjiw54OmCAU1irAHjlRoTd0WpoV6Qp1fXuIRU/edit#gid=1917070949`, 2023.

[5] G. Cocchi, A. Galligari, F. P. Nicolino, V. Piccialli, F. Schoen, and M. Sciandrone. *Scheduling the Italian National Volleyball Tournament*. Interfaces, 2018.

[6] W. Jang, H. H. Lim, T. J. Crowe, G. Raskin, and T. E. Perkins. *The Missouri Lottery Optimizes Its Scheduling and Routing to Improve Efficiency and Balance*. Interfaces, 2006.

[7] W. L. Winston. *Operations Research Applications and Algorithms Fourth Edition*. Curt Hinrichs, 2004.