# Optimizing Eta Kappa Nu Tutor Assignments

David Koplow
dkoplow@mit.edu

Raunak Chowdhuri
raunakc@mit.edu

December 8, 2023

## 1 Introduction

The Eta Kappa Nu honor society at MIT is the official tutoring service for all of MIT EECS. Currently, all pairings between tutors and tutees are done manually. This leads to inconsistent and suboptimal pairings due to the stochastic nature of when sign-ups occur, the availability of tutors, and the differing levels and areas of need for students. An optimization algorithm could create better pairings to meet all student demand. We reached out to the leadership of Eta Kappa Nu and were able to source anonymized historical data on student and tutor sign-ups. We use this data to build an optimization model that can be used to pair students and tutors.

## 2 Dataset

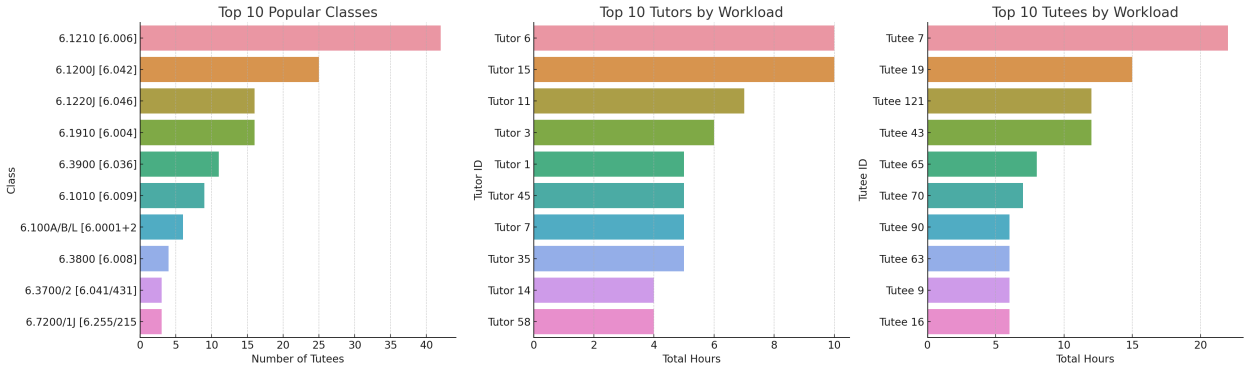


Figure 1: Fall 2023 Dataset Overview

The data consists of historical records of tutor and student sign-ups for tutoring sessions with Eta Kappa Nu. The dataset covers both Fall 2023 and Spring 2023. It includes the subjects students asked for help in, their performance at the time of request, as well as the availability of both the students and tutors along with the classes they were willing to teach. The data was acquired by one of the current Eta Kappa Nu Co-Presidents, who has access to records.

We performed some initial exploratory analysis with the datasets, visualized in Figure 1. We find that the classes students request tutoring in follow a pareto distribution, with select classes, like Introduction to Algorithms that request a bulk of the help. When examining tutor and tutee tutoring hours, however, we find a more even distribution with some outliers.

### 2.1 Challenges

Upon further exploration, we identified serious scheduling challenges with the existing dataset, as shown below in Figure 2. Some courses such as 53 lacked any tutors to meet the demand, while others like 6 required more tutoring than could be met by the amount tutors were expected to teach.

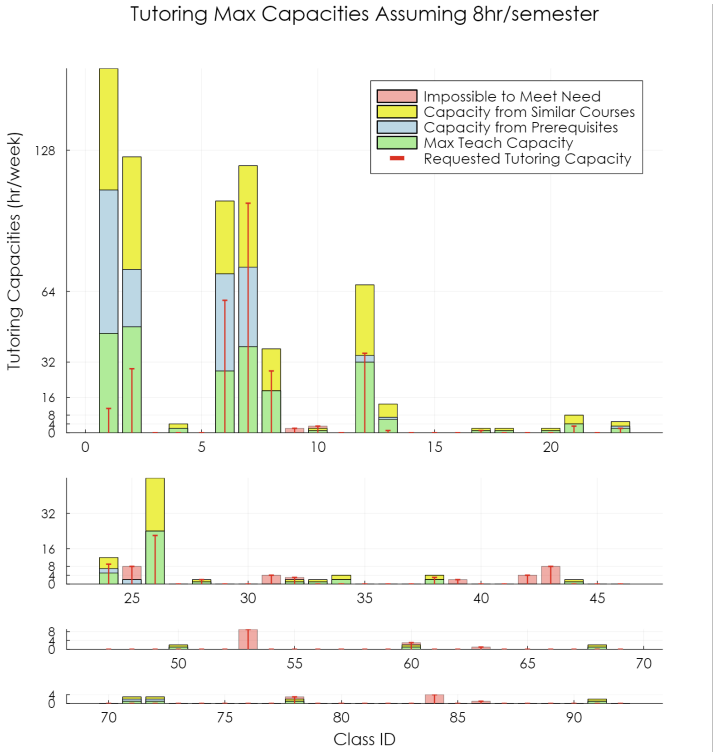


Figure 2: Fall 2023 Capacity

We note that there exist many classes for which students are marked as requesting tutoring, but no tutors are marked as being able to tutor. However, if we expand the set of classes that a tutor is able to tutor by examining prerequisites for the listed classes and looking at similar courses more students might be able to be served. This will motivate our use of a generative AI model to expand the set of classes that a tutor is able to tutor, described in Section 3.3

## 3 Methodology

### 3.1 Model 0: Naive Flow

To start, we built the simplest possible model for the problem. We created a linear variable representing how many hours a single tutor taught a single student in a single subject for:

$$x_{ijk} \geq 0 \, \forall i \in \text{Students}, j \in \text{Teachers}, k \in \text{Classes}$$

We then implemented the following constraints:
Teaching hours must not exceed the student's request:

$$\sum_{j,k} x_{ijk} \leq \text{Hours}_i \quad \forall i \in \text{Students}$$

Teacher's total teaching hours must respect their availability:

$$\sum_{i,k} x_{ijk} \leq \min(20, \text{Hours}_j) \quad \forall j \in \text{Teachers}$$

Teachers can only teach classes they are qualified for:

$$x_{ijk} = 0 \quad \forall i \in \mathit{Students} \quad \forall j \in \text{Teachers} \quad \forall k \notin \mathit{Classes}_j$$

**Objective Function:** Maximize the total teaching hours:

$$\max \sum_{i,j,k} x_{ijk}$$

This approach, as shown in Section 4, performs worse than the manual matching approach. When we look at the constraints above that the manual matching violates, we see that it is primarily respecting teacher availability constraints and teachers only teaching classes they are qualified for. This motivates our next step, which is to expand the set of classes that a tutor is able to teach.

### 3.2 Model 1: Weighted Optimization

We adjusted the previous model with some new settings: a mixed integer linear optimization model to maximize student tutoring coverage and tutor utilization. The decision variable $x_{ijc}$ indicates the weekly tutoring hours tutor $i$ spends with student $j$ for class $c$.

The objective maximizes weighted student tutoring coverage, but also considers minimizing the number of student tutor pairings and penalizing assignments that lead tutors to tutor more or less than 8 hours per week:

$$\max\ 10 \sum_{j,c} p_{jc} \sum_{i} x_{ijc} - \sum_{i} y^{<8} i + 2y^{>8} i - \sum_{ijc} n_{ijc}$$

Subject to constraints on:

- Tutor availability
- Maximum tutoring hours per student
- Mapping binary variables $y_i^{<8}$, $y_i^{>8}$ to penalize tutors who tutor less than 8 hours and encourage tutors who tutor more than 8 hours.
- Penalizing the total number of student tutor pairings $n_{ijc}$

The full model formulation is available in the appendix.

### 3.3 Generative AI Expansion

Motivated by the limited capacity to teach certain classes and its effect on fair tutor hour allocations, as observed in Figures 1 and 2 we utilized a large language model (LLM) from Anthropic to automatically expand the set of classes that a student tutor is capable of teaching. We mined the MIT course catalog and formatted the results in a way that the LLM could understand. Then, we repeatedly queried the model across each of the tutor's class selections to expand the set of relevant classes using the prompt described in 3. Then, we created an updated tutor-to-class mapping which was utilized by all of the subsequent models. We also constructed a set of prerequisites for courses tutors selected to tutor and allowed them to tutor those as well.

Here is a list of class descriptions: Class Descriptions
Here is a json of prereqs: Prerequisites
Question Start A student mentioned they are comfortable tutoring the following: List of Classes
'What other classes would they be capable of teaching?
Answer in the form of a Python list (on a single line).
Afterwards you may give justification. (e.g. ["6.100A"]) Response Start ["

Figure 3: Course list expansion prompt. Special tokens for designating prompt sections are marked in green, and external data which is substituted in is marked in red.

### 3.4 Model 2: Tutor Squads

The final model we built sought to overcome limitations in tutor availability by using the expanded tutor class selection described in 3.3 and allowing multiple students to be tutored together in one session. Even if each tutor taught a student for all available hours, we would not be able to match the demand for tutoring. To overcome this, we grouped students together into "squads" of up to 4 students who share tutoring interest in a given class. Instead of matching teachers to students directly, we matched teachers to our generated squads. This allowed us to match more students with fewer tutors, and also allowed us to match students who were not able to be matched in the previous model. The formulation of this model is shown in 6.2

## 4 Results

| **Model** | **Request (hr/w)** | **Taught (hr/w)** | **Taught (%)** | **Requests (#Stu.)** | **Some Tutoring (#Stu.)** | **Some Tutoring (%)** | **Tutor Time (hr±std)** |
|---|---|---|---|---|---|---|---|
| Manual (S23) | 621 | 186 | 29.95% | 170 | 96 | 56.47% | 2.7 ± 2.0 |
| Model 0 (S23) | 621 | 202 | 32.53% | 170 | 108 | 63.53% | 3.24 ± 3.0 |
| Model 1 (S23) | 621 | 202 | 32.53% | 170 | 58 | 34.12% | 2.89 ± 3.0 |
| Model 2 (S23) | 621 | 541 | 87.12% | 170 | 169 | 99.41% | 2.4 ± 3.0 |
| Manual (F23) | 366 | 170 | 46.45% | 133 | 105 | 78.95% | 2.3 ± 1.9 |
| Model 0 (F23) | 366 | 167 | 45.63% | 133 | 100 | 75.19% | 2.3 ± 2.6 |
| Model 1 (F23) | 366 | 167 | 45.63% | 133 | 68 | 51.13% | 2.3 ± 2.6 |
| Model 2 (F23) | 366 | 321 | 87.70% | 133 | 121 | 90.98% | 1.5 ± 1.7 |

Table 1: Model Results

We see that Model 2 almost doubles the performance when compared to manual matching while requiring significantly less work from the tutors. We also see that our automatic matching system leads to more students getting some tutoring than before. Models 0 and 1 generally perform about the same as manual matching with slightly more hours taught in S23, but slightly fewer in F23.

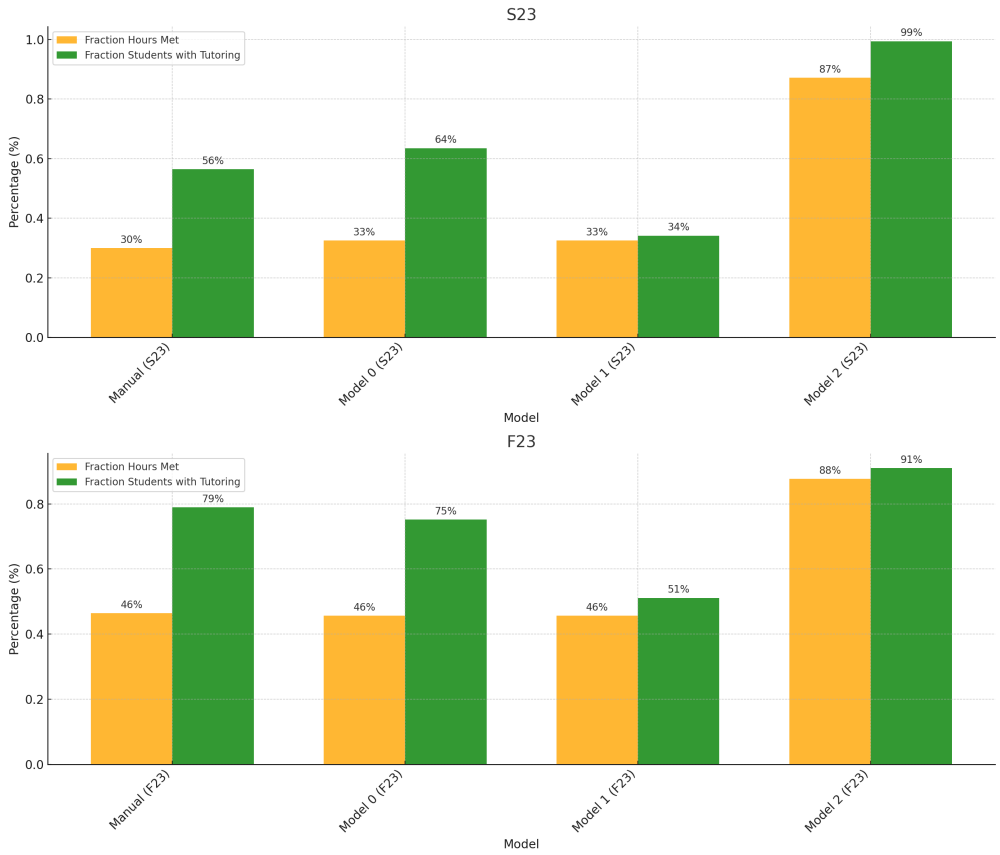


Figure 4: Model performance

Through inspecting the data it seems manual matching often violated the hour constraints listed in the data. The F23 performance difference is explained by the manual matching violating the hour constraint listed in the data. The sharp reduction in students with tutoring between Manual matching and Model 1 visible in Figure 4 are caused by the number of unique students with some tutoring not being a factor in the optimization. The significant spike we see for Model 2 is a result of just more students getting matched. Model 2 outperforms all other models in every metric.

# 5 Conclusion

This project will provide Eta Kappa Nu with an improved system for student-tutor pairing. The optimization approach described in Model 2 will be implemented as their actual pairing process in future semesters. This will make tutoring more accessible and effective for students. For students who need more individualized tutoring, they will still be able to request it. But now, HKN will meet about twice the demand it was able to before. The findings from this paper will help hundreds of MIT students access their education every year for years to come.

# 6 Appendix

All of the code is available on GitHub at https://github.com/raunakdoesdev/optimization-final-project.git, but this is a private repo, email raunakc@mit.edu to request access.

## 6.1 Model 1: Formulation

Given $r_{jc}$ the requested amount of tutoring per week of student $j$ for class $c$, $p_{jc}$ the need of student $j$ to get tutoring in subject $c$, and $t_i^{time}$ indicating a tutor's availability per week to tutor, and $t_{ic}^{class}$ indicating tutor i's ability to tutor in class $c$, the decision variable for the problem is $x_{ijc}$ indicating how many hours per week tutor $i$ meets with student $j$ about class $c$. Let $y_i^{<8}$ and $y_i^{>8}$ be binary variables representing if tutor $i$ has tutored more than or less than 8 hours over the course of the semester. Let $n_{ijc}$ be 1 if tutor $i$ teaches student $j$ class $c$. Finally, assume there are $w$ weeks in the semester, for this paper, we will assume there are 8 weeks of tutoring.

From the above description, we seek to maximize the objective:

$$\max \ 10 \sum_{j,c} p_{jc} \sum_i x_{ijc} - \sum_i y_i^{<8} + 2y_i^{>8} - \sum_{ijc} n_{ijc}$$

All models specified in this paper were optimized using Gurobi and will be evaluated to optimize pairings given constraints. Performance will be evaluated on student coverage, tutor utilization, and computation time. Cross-validation will tune parameters (weightings of different parts of the optimization problem), and evaluate performance.

Enforce positivity and binary variables $\forall i \ \forall j \ \forall c$:

$$0 \leq x_{ijc}, \quad x_{ijc} \in \mathbb{R}, \quad 0 \leq y_i^{>8}, y_i^{<8}, n_{ijc} \leq 1, \quad y_i^{>8}, y_i^{<8}, n_j \in \mathbb{N}$$

The student gets no more tutoring than requested: $\sum_i x_{ijc} \leq r_{jc} \ \forall c \forall j$
The tutor tutors no more than able: $\sum_c \sum_j x_{ijc} \leq t_i^{time}, \quad \sum_c \sum_j x_{ijc} \leq 20 \ \ \forall i$
Constraints to tie $y_i^{>8}$ and $y_i^{<8}$ to $x_{ijc}$: $\frac{8}{w} - M y_i^{<8} \ \leq \sum_c \sum_j x_{ijc} \leq \frac{8}{w} + M y_i^{>8} \ \forall i$
Constraints to tie $n_{ijc}$ and $x_{ijc}$: $x_{ijc} \leq M n_{ijc} \forall i \forall j \forall c$
Ensure no tutor tutors classes they aren't able to: $x_{ijc} \leq M t_{ic}^{class} \ \forall i \forall j \forall c$

## 6.2 Model 2: Formulation

We will modify the Model 1 in a couple of ways. First, we will change the objective function to include a different reward depending on whether the match comes from a class the tutor selected $t_{ic}^{class}$, a pre-requisite of one of those classes $t_{ic}^{pre}$, or a course with similar content $t_{ic}^{sim}$ for tutor $i$ and class $c$. By construction, if $t_{ic}^{class} = 1$, neither $t_{ic}^{pre}$ or $t_{ic}^{sim}$ will equal 1. Additionally, if $t_{ic}^{pre} = 1$, then will $t_{ic}^{sim} = 0$. Since matching is done in squads, $r$ is replaced with $r_{jc}^{squad}$ which is 1 if the squad requests tutoring for the class and 0 otherwise. One final note, here j represents a squad of up to 4 students that need to be tutored. The new objective function:

$$\max \ \sum_{j,c} p_{jc} \sum_i x_{ijc}(10 t_{ic}^{class} + 8 t_{ic}^{pre} + 5 t_{ic}^{sim}) - \sum_i y_i^{<8} + 2 y_i^{>8} - \sum_{ijc} n_{ijc}$$

.

Additionally, we must modify the constraint responsible for constraining $x_{ijc}$ to the classes the tutor can teach and the requested hours of a student (now squad):

$$x_{ijc} \leq M(t_{ic}^{class} + t_{ic}^{pre} + t_{ic}^{sim}) \ \forall i \forall j \forall c, \quad \sum_i x_{ijc} \leq r_{jc}^{squad} \ \forall c \forall j$$