Chapter 2

Formalizing the Problem

2.1 Introduction

The existing research on school choice and course allocation would seem to suggest that for our problem, which only features one-sided preferences as opposed to preferences on both sides of the market, we follow a fair division approach similar to that of Budish's one-sided matching [1]. However, given the practical application of our problem as a mechanism for assigning every student to one specific class, we determined that constructing a notion of stability makes more sense than evaluating envy in the matching. Our solution is driven by the goal of assigning HUM classes that are balanced both in size and student interests, so we want to minimize the desire of students to swap with each other post-assignment as much as possible so as not to disrupt the balance. If no student wishes to swap with another student, then we can consider the assignment stable. This is the framework with which we shall formalize our model.

2.2 The Model

The basic components of our model are the two sides of our market: a set of students and a set of HUM class sections, each of which have a certain number of seats. Though we are working in a bipartite setting, only one side of our market has preference over the other (i.e. students prefer certain classes, classes are inanimate and therefore indifferent to students...as far as we know). Students also do not necessarily have ranked preference over every single available section; they likely equally do not want to take any sections that conflict with their other selected courses. So, preference lists will be incomplete, with those that students have a conflict with not being included in their ranking.

Definition 2.1. The *HUM-assignment* problem consists of a finite set of students, S, and a finite set of class sections C. Each class section $c \in C$ has a capacity q_c representing the number of available seats in the section. Each student $s \in S$ has a strict preference relation $\succ_s$ over $C \cup \{\emptyset\}$, where $\emptyset$ represents the outside option of a student such that $\emptyset \succ_s c$ for some $c \in C$ denotes that the class is not an option for the student.

Definition 2.2. A *1-sided matching (1M)* of students to class sections is a function $\mu : S \to C \cup \emptyset$ such that

- Each student is assigned to either a class they prefer or no class: $\mu(s) = \emptyset$ or $\mu(s) \succ_s \emptyset$ for all $s \in S$

- No class exceeds its capacity: $|\{s \in S : \mu(s) = c\}| \leq q_c$ for all $c \in C$

Finding a 1-sided matching would solve an instance of the HUM-assignment problem as defined above. However, we still have to introduce the last constraint of the problem: *proportionality*.

Students can be divided into three types based on the way Reed College divides majors into groups: Group 1 for language, arts, and literature, Group 2 for social sciences, Group 3 for math and lab sciences. To implement the HUM philosophy, which is that the class provides an opportunity for students of all knowledge backgrounds and academic interests to discuss relevant texts, we want to ensure a balanced spread of each type of student between sections. This means avoiding scenarios such as all chem 101 students with the same lab being placed in the same section due to the overwhelming similarity in their schedules. This will come in the form of quotas or caps on the number of seats per section that can be assigned to each type of student, so each class will be divided evenly into four types of available seats: A, B, C, and free seats to allow for some flexibility.

Definition 2.3. Under the proportionality constraint, S is partitioned into three sets: $A, B, C \subset S$, where $A \cup B \cup C = S$ and $A \cap B \cap C = \emptyset$. Let students $s_1, s_2 \in A$ be denoted a_1, a_2.

Definition 2.4. For each class c with capacity q_c, let $q_c^t = \lfloor q_c/4 \rfloor$ for each $t \in \{A, B, C, F\}$, where A, B, C correlate to the types of students and F represents the set of free seats.

Definition 2.5. A *proportional 1-sided matching (P1M)* builds on our definition of 1-sided matching by adding the quality that no class exceeds its capacity for each type of student: $|\{s \in A : \mu(s) = c\}| \leq q_c^A + q_c^F$.

To make things clearer, we can represent our problem as a graph to provide us with a visualization of the problem (see Figure 2.1).

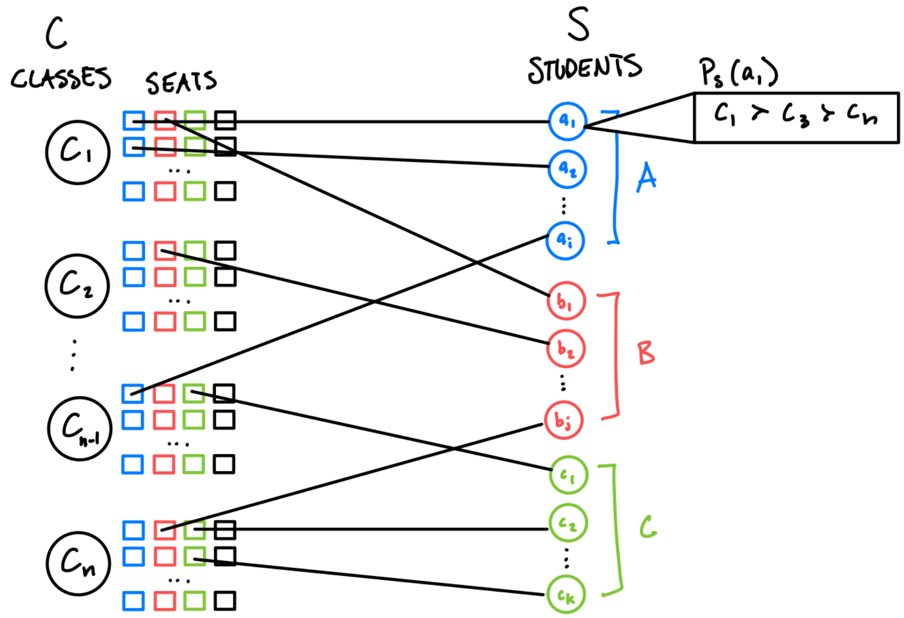

Figure 2.1: Graph illustrating a proportional 1-sided matching

In a graph representing the HUM-assignment problem, the vertices are comprised of students and classes, and an edge is drawn between a student and a class if the class is on the student's preference list. We can formally define this representation.

Definition 2.6. Let $G = (V, E)$ be a bipartite graph where:

- $V = S \cup C$
- $E = \{(s, c) \mid s \in S, c \in C, c \succ_s \emptyset\}$

In this form, the graph only represents our problem, not a solution. For a graph to represent a solution or a matching, each student would only have one edge connecting them to a single class in a way that adheres to class capacities. So, we need a way to narrow down which edges lead to a viable solution. That means we need a way to judge edges against each other: we need to assign value to them. To accomplish this, we can give each edge a weight that represents how conducive it is to the constraints of our problem. Recall that we want to find a solution that satisfies two goals: every student is assigned to a class they prefer, and all reserved seats are filled by students of that reserved type. We can assign weights to the edges based on these goals such that a higher weight indicates that edge embodies more of the qualities we want in our solution. Because we have two goals that are equally important, we can simply increment the weight of each edge by 1 for each goal it satisfies. We can now supplement our previous definition of a graph.

Definition 2.7. Let $G = (V, E)$ be a bipartite graph where:

- $V = S \cup C$
- $E = \{(s, c) \mid s \in S, c \in C, c \succ_s \emptyset\}$
- $w : E \to \{1, 2\}$ is a weight function defined as

$$w(s, c) = \mathbf{1}\{\text{if student } s \text{ prefers class } c\} + \mathbf{1}\{c \text{ is reserved for } s\text{'s type}\}$$

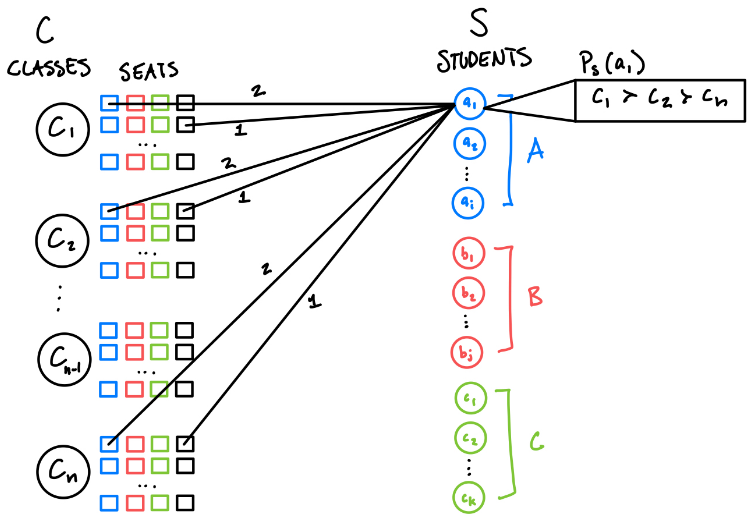

Figure 2.2: Graph illustrating weighted edges for one student

Now, we have a way to compare edges to each other to allow us to find a solution. Based on the way we've set up our weights, we know that the best available solution will be the graph with the maximum possible weight when only one edge remains for each student. Our max-weight matching then has the properties required by our P1M definition, being that students of each type are divided relatively evenly to classes they prefer to be in, as well as maximizing the overall satisfaction of students with their assignments.

Definition 2.8. A matching μ is a *maximum-weight matching (MWM)* if for all P1M matchings μ':

$$\sum_{s \in S} w(s, \mu(s)) \geq \sum_{s \in S} w(s, \mu'(s))$$

Observe that a maximum-weight matching would not only serve as a P1M solution, but it would also (by nature of the way we assign weights based on student preferences) ensure the solution is one in which the greatest number of students are happy with their match. So, viewing the problem through the lens of a weighted graph allows us to both solve the problem, and provide the solution that gives the most student satisfaction.

2.3 The Challenge of Stability

We've determined how to find a solution to the problem we defined, and even find a solution that maximizes student happiness, but there is one more aspect to the problem we must consider. Recall that matching is often studied under the analysis of **stability**. Notably, "stability" in our case adopts a different meaning than in a traditional stable matching problem. Because we are working with one-sided preferences, we do not consider "cheating" as a violation of stability between students and classes, but rather between students themselves. That is, we can consider a matching stable if there are no blocking pairs *between students*.

Definition 2.9. A matching μ is *stable* if there exists no blocking pair between students $s_1, s_2 \in S$ such that $\mu(s_2) >_{s_1} \mu(s_1)$ and $\mu(s_1) >_{s_2} \mu(s_2)$.

This notion of stability proves difficult for our problem, however. If we look back at our definition of the HUM-assignment problem with class capacities and incomplete preferences, we will find that in some instances it is possible there may not even exist a stable matching to find.

Theorem 2.10. *There is not always a stable matching with quotas and incomplete preferences.*

Proof. Let $C = \{c_1, c_2\}$, where there are two types of student A and B such that for each $c \in C, q_c^A, q_c^B = 1$ (each class can have at most one student of each type). Let $S = \{a_1, b_1, a_2, b_2\}$. Let the preference lists be as follows:

$$\begin{aligned}
a_1 &: \quad c_1 \\
b_1 &: \quad c_2 \\
a_2 &: \quad c_1 \succ c_2 \\
b_2 &: \quad c_2 \succ c_1
\end{aligned}$$

In this case, regardless of the order of matching or algorithm, there is no stable matching. The A students will both wish to be matched to c_1, which only has one A spot, and both B students wish to be matched to c_2 which only has one B spot. So, when the first round of matching is done, that means that there is still an open B spot for c_1 and an open A spot for c_2, which the unhappy A and B students can then switch into on their own, respectively. $\square$

Defining stability in this way is, then, too strict. Recall that even determining whether there is a stable matching or not is NP-complete in many scenarios similar to this one [2]. So, we need to relax our constraints to allow us to explore some other notion of stability. For now, let's simplify the problem. At a basic level, before we even think about stability, we want to work in an environment where we can be sure an algorithm would respect student preferences, such that no student is assigned to a class they did not list as an option.

To temporarily remove our notion of stability, we can alter our preferences from a ranked system to what is known as **0/1 preferences**. In 0/1 preferences, students assign a class a 1 if they are okay with it and a 0 if it is not an option for them. This way, there cannot be a notion of stability because all preferred classes have the same rank. Now, for ease of notation, we will let C_r be the set of reserved seats across all classes. Notably, this simplified environment still lends itself well to the same weighted graph construction from definition 2.7, so we can still use it to represent our problem.

Theorem 2.11. *If a matching that respects student preferences and seat reservations exists, it will be the maximum-weight matching where* $w(\mu) = |S| + |C_r|$.

Proof. Let μ be a matching that assigns every student to a seat they prefer while also filling reserved seats first, such that all reserved seats are filled before the overflow seats. By construction of the weights, we can determine a lower and upper bound for $w(\mu)$.

1. **Lower bound of μ:** Each edge exists only if the student prefers that seat, so every edge in μ contributes at least 1. By our assumption of characteristics of μ, every student in μ got a seat they prefer, so summing over all students contributes at least $|S|$ to $w(\mu)$. We also assumed each reserved seat was assigned to a student in μ, which adds an extra $+1$ to the weight for the edge from each of those reserved seats. Summing over all reserved seats gives an additional contribution of $|C_r|$, because the rest of the seats are still 1-weight edges. Therefore, we can bound μ:

$$w(\mu) \geq |S| + |C_r|.$$

2. **Upper bound of μ:** From our definition, we know each edge weight is at most 2. Similarly to the lower bound, we know μ has $|S|$ edges because each student was matched to one of their preferences. However, because only the reserved seats contribute an additional weight to an edge, we know that the number of 2-weight edges $\leq |C_r|$. So, the maximum summed weight of all edges must be $2|C_r| + |S| - |C_r|$, representing the maximum number of 2-weight edges plus all students assigned to an overflow seat. Note that is equivalent to $|S| + |C_r|$. Therefore, we can bound μ:

$$w(\mu) \leq |S| + |C_r|.$$

So, μ is a maximum-weight graph with a lower bound of $|S| + |C_r|$ and an upper bound of $|S| + |C_r|$, so it must be that a matching that fully respects student preferences and seat reservations has a weight $w(\mu) = |S| + |C_r|$. □

Theorem 2.11 shows us that using an algorithm that finds the maximum-weight matching, we can assign each student to a class they prefer while respecting proportionality constraints if such a matching exists. From here, we can begin to reintroduce stability. While it may not be feasible to find stability between every student, perhaps we can start by comparing just students of the same type. We are introducing a new notion of stability here, where we will consider a matching stable if it exhibits **intra-group stability**. This means that for any type of student, that student does not form a blocking pair with another student **of the same type**.

Definition 2.12. A matching μ is *intra-group stable* if, for any two students t_1 and t_2 for all types $T \subset S$, $\mu(t_1) >_{t_1} \mu(t_2)$ or $\mu(t_2) >_{t_2} \mu(t_1)$ such that there exist no intra-group blocking pairs.

With our new notion of intra-group stability, we can now add to our understanding of what makes a viable solution to our HUM-assignment problem.

Definition 2.13. An *intra-group stable proportional 1-sided matching (ISP1M)* is a function $\mu : S \to C$ such that

- Each student is assigned to a class they prefer:
  $\mu(s) \succ_s \emptyset$ for all $s \in S$
- No class exceeds its capacity: $|\{s \in S : \mu(s) = c\}| \leq q_c$ for all $c \in C$
- No class exceeds its capacity for each type of student:
  $|\{s \in T : \mu(s) = c\}| \leq q_c^T + q_c^F$
- There exist no intra-group blocking pairs:
  for all $t_1, t_2 \in T \subset S$, $\mu(t_1) >_{t_1} \mu(t_2)$ or $\mu(t_2) >_{t_2} \mu(t_1)$

For now, our goal will be to develop an algorithm that finds an ISP1M to solve our problem. We will classify our previous definition of stability (definition 2.9) as **inter-group stability** (blocking pairs involving students of different types) to draw a distinction between the two notions. For now, inter-group instability does not interfere with our search for an intra-group stable solution. Though we would ideally guarantee stability between all students, looking at the problem through the lens of intra-group stability allows us to give some theoretical guarantee about minimizing the amount of swapping that will occur in an assignment while remaining computationally feasible, as opposed to turning to less-reliable heuristic solutions to try to solve the broader conception of the problem.