The University Course Assignment Problem : An Application of Graph Theory

Rishabh Khandelwal (2021B3A71207G)
Dishant Surana (2021B3A71114G)
Mehak Garg (2022A7PS0803G)

November 30, 2023

1 Objective

To formulate various assignments of courses to professors adhering to their desired workloads and aligning with their respective preference orders, while also making sure that the minimum requirement for CDC's and electives is fulfilled for both First Degree (FD) as well as Higher Degree (HD)

2 Constraints

1. No professor should work below his/her desired workload capacity in an ideal scenario. Exception- There could be a case where the total combined workload of the professors is a fractional number, in such a case we have to reduce the workload of a single faculty by 0.5 so that the workload becomes a whole number.
2. Assignment of courses should proceed strictly aligning with the preferences provided by the professors. Courses which do not appear on a professor's preference list cannot be allotted to him/her.
3. All the CDC's (FD and HD) should be compulsorily offered.
4. The minimum requirement for Electives should also be fulfilled. (2 each for FD and HD)

3 Algorithm Used: Minimum Weight Full Matching Algorithm

A matching or independent edge set in an undirected graph is a set of edges without common vertices. The Minimum weight full matching algorithm is available as a function in the NetworkX library in Python. It computes a matching with

the minimum sum of weights assigned to various edges of a bipartite graph by selecting an appropriate assignment. The matching (set of edges) output by the algorithm has a cardinality equal to that of the smaller of the two bipartite sets.

4 Methodology Adopted

We consider the problem given to us as a bipartite graph where professors are mapped to courses. A matching returns a set of independent edges so it cannot map one professor node to multiple course nodes. So we construct the graph such that each course is represented by two nodes (each node representing half), and each professors of type x1, x2 and x3 are represented by 1, 2 and 3 nodes respectively (each node corresponding to 0.5 course load). Now we have to assign weights to each edge as per the preference orders. The edge corresponding to the course at the 1st priority is assigned a weight of 1, similarly for the course at 2nd priority the weight is 2 and so on. We now have a bipartite graph which has weighted edges between each professor to and each course which appears in his preference list.

After applying the modifications mentioned below, the graph is now ready to be directly plugged into the weight minimisation algorithm. Since modifications have been made such that the total workload is exactly equal to the minimum (or more if possible) courses that need to be offered, our bipartite graph will have exactly the same number of vertices in either bipartite set. This is because every professor node corresponds to 0.5 courseload taken by him, and every course node corresponds to 0.5 of the course. The algorithm would find an assignment which minimises the sum of the weights assigned to each edge. In our case, it means that the assignment would perfectly align with the preferences given by the professors. Since there is no prioritization among professors, it is best to try and maximize the collective utility of all professors (which in our case is done by minimizing total weight because a lower weight corresponds to a higher position in the professors preference list). Since the constraints have been handled separately by crash tests and fixing the degrees of all the vertices, the assignment(s) given by the algorithm would be the most optimal combination(s) possible. Our modifications also ensure that every professor works at exactly his desired workload.

The above method considers CDCs and electives to be of equal priority during the allotment process (while obviosuly ensuring that all CDCs are offered). An alternate allotment is provided as well. This is computed using the exact same methodology as above, but with a slightly different method of assigning weights to edges, such that we can prioritize CDCs over electives during allotment. Edges corresponding to CDCs are allotted weights equal to their position in the preference list, but for electives, the position is multiplied by an arbitrary number(say 2) to deliberately assign them a higher weight. The electives will still be assigned, but because of the higher weight, the algorithm will allot CDCs first, and then move onto CDCs, yielding a different allotment from the previous case.

4.1 Modifications Made

1. Since there is a constraint in the given algorithm that the degree of each vertex has to be 1, we assume that each edge corresponds to 0.5 course allocation. So, for x3 type professors(for eg. prof10), we will denote each professor by 3 vertices- prof10A, prof10B and prof10C, meaning that he/she can be assigned 1.5 courses per semester. Similary for x2 type professors(for eg. prof5), there would be two vertices - prof5A, prof5B.

2. If the cumulative workload of the professors exceeds the minimum course requirement (CDCs + 4 electives), extra electives are chosen from the pool of FD electives, till the total workload is satisfied. The additional FD electives for this purpose are chosen on the basis of popularity (i.e, how many different peoples preference list a particular course appears on)

3. If the total workload is not a whole number, there must be a professor who will teach below capacity even after increasing the electives (because a course being offered must be taught completely, cannot have only 0.5 of the course assigned to a prof). In this project, we are reducing the workload of a random professor of x3 category (from 1.5 courses to 1).

5 Crash Tests

1. Total workload is less than min courses (CDCs + 4 electives), i.e., there are not enough professors to teach the minimum required courses.

2. Any prof has not filled the required minimum number of preferences for one or more of the course categories.

3. The number of courses (filled in all preferences collectively) of each different category is less than the minimum no. of courses for that category, i.e not enough professors prefer to teach courses of that category.

4. A unique course is defined as a course which only appears once in all the preferences combined. If a unique course appears in the preference list of a professor of x1 category, no assignment is possible as his/her capacity is just 0.5 courses. Furthermore, there would be a crash test even if 2 unique courses appear in the preference list of the same professor, as it is impossible to allot both courses to that professor (maximum capacity of a professor is only 1.5 courses.)