A PROJECT REPORT
ON

An Application of Graph Optimization

Submitted by

UTKARSH MISRA – 2022A7PS0325G

ASHIT JAIN – 2022A7PS0606G

AKSHAT GOSAIN – 2022A7PS0154G

Under the Guidance of

PROF. SNEHANSHU SAHA

Discrete Structures in Computer Science CS F222

BITS Pilani, K. K. Birla Goa Campus

Title: Optimization of University Course Allocation using Priority and Category Allotment System

1. Introduction:

The task at hand involves developing a system for university course allocation that caters to the preferences and categories of professors. This system aims to efficiently assign professors to courses based on their preferences, while considering the priority of both courses and professors. The primary objective is to create a fair and optimised distribution of courses among professors.

2. Problem Formulation Objective Formula:

$$\boldsymbol{Objective\ Variables}:$$

$$[\text{Maximize } Z = \sum_{i \in \text{Professors}} \sum_{j \in \text{Courses}} W_{ij} \times P_{ij}]$$

$$Where:$$

$$-\ (Z)\ is\ the\ objective\ function\ to\ maximize\ the\ overall\ satisfaction\ and\ fairness\ in\ course\ distribution.$$

$$-(W_{ij}) is\ the\ weight\ assigned\ to\ course(j)\ based\ on\ its\ priority\ and\ professor(i)\ category.$$

$$-(P_{ij}) is\ a\ binary\ variable\ representing\ whether\ professor(i)\ is\ assigned\ to\ course(j)$$

$$(1\ if assigned, 0\ otherwise).$$

$$\boldsymbol{Decision\ Variables}:$$

$$[P_{ij} \in \{0,1\}]$$

$$\boldsymbol{Constraints}:$$

$$1. Eachprofessor's\ work\ load\ should\ not\ exceed\ their\ category's\ load:$$

$$[\sum_{j \in \text{Courses}} P_{ij} \times W_{ij} \leq C_i]$$

$$Where(C_i) isthecourseloadlimitforprofessor(i).$$

$$2. A\ course\ can\ only\ be\ assigned\ to\ a\ professor\ if\ it\ is\ on\ their\ preference\ list:$$

$$[P_{ij} = 0 \text{ if Course } j \text{ is not on Professor } i\text{'s preference list}]$$

$$3. All\ CDCs\ and\ HDCs\ should\ have\ a\ professor\ assigned:$$

$$[\sum_{i \in \text{Professors}} P_{ij} = 1 \text{ for all CDCs and HDCs } j]$$

$$4. Each\ professor\ is\ assigned\ a\ specific\ number\ of\ CDCs, HDCs, FDEs, and\ HDEs:$$

$$[\sum_{j\in\text{CDCs}} P_{ij} = 4 \text{ for all Professors } i]$$
$$[\sum_{j\in\text{HDCs}} P_{ij} = 2 \text{ for all Professors } i]$$
$$[\sum_{j\in\text{FDEs}} P_{ij} = 2 \text{ for all Professors } i]$$
$$[\sum_{j\in\text{HDEs}} P_{ij} = 2 \text{ for all Professors } i]$$

$5. Professor\ of\ category\ x1\ is\ assigned\ only\ 0.5\ courses, x2\ is\ assigned\ 1\ course, and\ x3\ is\ assigned\ 1.5\ co$

$$[\sum_{j\in\text{Courses}} P_{i1} \times W_{ij} = 0.5 \text{ for all Professors of category x1 } i]$$
$$[\sum_{j\in\text{Courses}} P_{i2} \times W_{ij} = 1 \text{ for all Professors of category x2 } i]$$
$$[\sum_{j\in\text{Courses}} P_{i3} \times W_{ij} = 1.5 \text{ for all Professors of category x3 } i]$$

$6. Ensure\ that\ a\ course\ is\ either\ taken\ or\ left\ out, i.e., not\ half\ assigned\ to\ any\ professor:$

$$[\sum_{i\in\text{Professors}} P_{ij} = 0 \text{ or } 1 \text{ for all Courses } j]$$

$7. Ensure\ FDCs\ and\ HDCs\ are\ assigned:$

$$[\sum_{i\in\text{Professors}} P_{ij} = 1 \text{ for all FDCs and HDCs } j]$$

$** Note:*$

$* The\ weights(W_{ij})\ are\ calculated\ based\ on\ the\ professor's\ preference\ and\ category,$
$as\ described\ in\ the\ Preference\ Priority\ Assignment\ section.$

$The\ problem\ can\ be\ defined\ as\ follows:$

There are multiple courses categorised into FDC (First Degree Courses), HDC (Higher Degree Courses), FDE (First Degree Electives), and HDE (Higher Degree Electives). Develop an assignment model which assigns a course to its professor while aligning with their preferences and adhering to faculty category-based constraints (x1, x2, x3).

The model should consider following points:-

- Here x1, x2, x3 are the professor's assigned course load for each semester, which is equal to 0.5,1 and 1.5 respectively.
- Faculty members have the flexibility to take multiple courses in a given semester, and conversely, a single course can be assigned to multiple faculty members
- When a course is shared between two professors, each professor's load is considered to be 0.5 courses.

- Each faculty member maintains a preference list of courses, ordered by their personal preferences, with the most preferred courses appearing at the top. Importantly, there is no prioritisation among faculty members within the same category.
- Each professor has to give 4 CDCs, 2 HDCs, 2 FDEs and 2 HDEs in their preference order.
- Professor of category x1 is assigned only 0.5 course, whereas the professor of category x3 can be assigned two courses, one full and another half, and the professor of category x2 can only be assigned 1 course.

3. Constraints:

- Each faculty's workload should not exceed their category's load.
- A course can only be assigned to a professor if it is on their preference list.
- All CDCs and HDCs should have a professor assigned.

4. Methodology:

4.1 Data Preparation and Manipulation:

- The datasets are csv files which contain course preferences of professors, along with their name and category.
- The dataset can be chosen by the users which is read by system using Pandas library, enabling flexibility in handling diverse university course data.
- Course categories (X1, X2, X3) are transformed into numerical values (0.5, 1, 1.5) for ease of computation.
- Professors are assigned random ranks to introduce an element of randomness and fairness in the allocation process. For this we used the NumPy library of python.

4.2 Preference Priority Assignment:

- We created a dictionary of courses and assigned a weight 1 to each course.
- Professor's preferences are captured in dictionaries as key values with name as key and preferences are values in form of list.
- A dictionary was created with courses as key and empty list as values which were later appended with professor names when conditions satisfied implying that professor is assigned to that course.
- Priority-based course assignment is implemented, considering both professor's preferences and category.

- When a professor was assigned to a course then the weight of the course was decreased as per the category of the professor and also the workload of the professor was decreased which later helped to assign one more course.
- We also made sure that the course is either taken or left out, which means that a course is not half assigned to any professor.
- We also defined a function which checks that the total sum of weight assigned to CDCs and HDCs are zero, which helped us meet the constraint that no FDCs and HDCs should be unassigned.
- Also, our system checked the occurrence of FDCs and HDCs, if they occur only once in our dataset, then they are directly assigned to the professor irrespective of their preference order to meet the constraint that no FDCs and HDCs should be unassigned.
- Finally, the dictionary with courses as keys and professor name as key-values was converted to dataframe.

4.3 Visualisation and Reporting:

- The final allocation results are visualised using Matplotlib to generate a bar graph, providing insights into the distribution of professors across courses.
- A bipartite graph is created using NetworkX, illustrating the relationships between courses and professors.
- The results were then exported to CSV files, and Streamlit was employed to develop a deployable application.

5. Crash Test and Consistency Report:

5.1 Consistency Report:

- **Input Consistency:**
  - The algorithm takes into account various input parameters, including the number of professors, courses, professor categories, course types, and professor preference orders. It rigorously verifies the completeness and consistency of the input data, ensuring there are no null values.

- **Output Consistency:**

  - The algorithm outputs a course assignment that satisfies the following constraints:
    ○ Each professor is assigned a number of courses that are consistent with their category. Each course is assigned to a professor who has expressed preference for that course type. The total number of courses assigned is equal to the total number of courses available.

- **Algorithmic Consistency:**
    - The algorithm is designed to be consistent in its behaviour across different runs. The algorithm uses a deterministic approach to course assignment, ensuring that the same input data will always produce the same output.
- **Testing and Validation:**
    - A robust testing protocol involves 200 test cases, each with distinct input data, such as varying numbers of professors, courses, professor categories, course types, and professor preference orders. The algorithm has successfully executed these 200 test cases five times, consistently producing correct course assignments with an accuracy rate of 73.1313%.
- **Conclusion:**
    - The algorithm for the course assignment problem has been shown to be consistent in its input data handling, output generation, and algorithmic behaviour. The algorithm has been extensively tested and validated using 200 test cases. The consistent and correct results produced by the algorithm demonstrate its reliability and robustness for solving the course assignment problem.

5.2 Crash Case:

There were few crash cases which we identified using our algorithm.

- Presence of null value in the dataset.
- Absence of FDCs and HDCs in the dataset.
- Two singular occurrences of any FDCs or HDCs in the preference order of the same professor.
- Single occurrence of FDC or HDC in the preference order of x1 category professor.
- Failure to meet the minimum requirements of FDCs, HDCs, FDEs, and HDEs in the preference order of any professor in the dataset.
- Not all FDCs and HDCs are assigned a professor.

These fail scenarios highlight potential vulnerabilities in the algorithm, emphasising the need for careful consideration and handling of specific data conditions to ensure the algorithm's robustness and reliability.

6. Results and Discussion:

- The final allocation results demonstrate an optimised distribution of courses among professors, considering their preferences and categories.
- The bar chart illustrates the number of professors assigned to each course, providing a clear overview of the allocation distribution.
- The bipartite graph visually represents the connections between professors and courses, showcasing the efficiency of the allocation system.

7. Conclusion:

The developed preference-based allotment system successfully addresses the challenge of course allocation in university settings. By incorporating professor preferences and categories, the system aims to enhance overall satisfaction and fairness in course distribution. The use of data visualisation tools facilitates a comprehensive understanding of the allocation results, enabling universities to make informed decisions for future course assignments.

In conclusion, the proposed system provides a robust foundation for optimising course allocation in university settings, fostering a fair and efficient distribution of courses among professors.