Project Report on

University Course Assignment System

An Application of Graph Optimization

By

Somanshu Rath (2022A7PS0032G)

Sanskar Mundhra (2022A7PS0757G)

Aditya Sharma (2022A7PS1108G)

For Course CS F222 DISCO (Semester III)

For Academic Year 2023-2024

Under the Guidance of

Prof. Snehanshu Saha

Dept. of Computer Science and Information Systems

BITS Pilani, K.K. Goa Birla Goa Campus

Problem formulation

Problem Statement

The research problem at hand focuses on optimizing the University Course Assignment System (UCAS). Within a department, there exist "n" faculty members categorized into three distinct groups: "x1," "x2," and "x3." The course load assigned to faculty members varies based on their category: "x1" faculty handle 0.5 courses per semester, "x2" faculty take 1 course per semester, and "x3" faculty manage 1.5 courses per semester.

The UCAS allows flexibility for faculty members 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. Additionally, each faculty member maintains a preference list of courses, ordered by their personal preferences, with the most preferred courses appearing at the top. Notably, there is no prioritization among faculty members within the same category.

The primary objective of this research is to develop an assignment scheme that maximizes the number of courses assigned to faculty while adhering to their preferences and the category-based constraints ("x1," "x2," "x3"). The challenge lies in ensuring that a course can only be assigned to a faculty member if it is present in their preference list.

Objectives

The primary objectives of the University Course Assignment System are to:

a. **Maximize Course Assignments:** Optimize the allocation of courses to faculty members, ensuring the highest possible number of courses are assigned while respecting constraints.

b. **Adhere to Faculty Preferences**: Prioritize faculty preferences by assigning courses that align with their individual interests and expertise.

c. **Address Capacity Constraints**: Ensure that course assignments comply with the workload capacities of each faculty category. Faculty members in categories "x1," "x2," and "x3" have distinct course load expectations, and the system must consider these limitations while assigning the courses.

Methodology

Approach Overview

Heuristic Approach:

The proposed heuristic approach employs randomization of the professor assignment order and a greedy algorithm that attempts to assign the most suitable course to each professor. This process is iterated, with each iteration evaluating the compatibility of potential course assignments for each professor, considering both the professor's available slots and the course's remaining capacity. If an assignment is compatible, it is finalized.

Due to the inherent non-optimality of this approach, the implementation was chosen to be in C++ for its efficiency. The program reads input data from a file named "input.txt" and stores the necessary information in variables and vectors. To facilitate data manipulation, two classes, namely Course and Prof, were created. Subsequently, deep copies of these vectors are passed to the solve function, after the Profs vector has been shuffled using the random_shuffle() function. We apply the greedy algorithm within the solve function and store the solution only if its unique.

Hungarian Algorithm Approach:

The Hungarian algorithm, also known as the Kuhn-Munkres algorithm, is a combinatorial optimization algorithm that solves the assignment problem in polynomial time. The assignment problem is a classic problem in optimization that seeks to find the optimal assignment of a set of tasks to a set of agents, such that the total cost or effort is minimized.

The Hungarian algorithm is based on a concept called a "matching." A matching is a set of edges in a bipartite graph, such that no two edges share a vertex. A matching is considered "perfect" if it includes exactly one edge for each vertex in one of the partitions of the bipartite graph.

The Hungarian algorithm works by iteratively improving a matching until a perfect matching is found. The algorithm starts with an initial matching, which can be any matching. Then, the algorithm iteratively updates the matching by finding an augmenting path, which is a path in the graph that starts at an unmatched vertex and ends at another unmatched vertex. The augmenting path is used to update the matching by switching the edges on the path.

The Hungarian algorithm is guaranteed to find the optimal matching in polynomial time, which means that the algorithm's running time is bounded by a polynomial function of the size of the input. This makes the Hungarian algorithm a very efficient algorithm for solving the assignment problem.

We implemented the Hungarian Algorithm as per our requirement and modelled the code according. It took input from the same input.txt file and with various functions pertaining to the Hungarian Algorithm, the program was able to return the most optimal assignments in the terminal.

Comparative Analysis

- Optimality: The Hungarian Algorithm output typically demonstrates optimality, providing assignments that maximize course allocations based on preferences and constraints.

- Heuristic Approach: While not guaranteed to be optimal, the heuristic approach output offers diverse solutions, potentially exploring a wider solution space than the Hungarian Algorithm.

Documentation on the results under different test cases

Test Case 1: 7 Faculty Members, 6 Courses

Scenario Overview:

Faculty Members: A, B, C, D, E, F, G

Courses: DSA, DBS, MPI, CN, DAA, CC, ML

Course Load Categories: 0.5, 1, 1, 1.5, 1.5, 1.5

Observations - Hungarian Algorithm (Python):

Optimal Assignments: The Hungarian Algorithm efficiently assigned courses to faculty members while considering their preferences and workload capacities. Faculty Preferences: Evident from the assignments, each faculty member received courses aligning with their preferences. Category-Based Constraints: Categories (0.5, 1, 1, 1.5, 1.5, 1.5) were respected, ensuring workload capacities were balanced across faculty members.

Observations - Heuristic Approach (C++):

The Heuristic Approach presented efficient assignments but lacked optimality. Prioritization was observed, albeit not globally optimal, providing insight into trade-offs between efficiency and optimality. The approach showed robustness in handling assignments for a moderate dataset size.

Test Case 2: 15 Faculty Members, 12 Courses

Scenario Overview:

Faculty Members: A, B, C, D, E, F, G, H, I, J, K, L

Courses: DSA, DBS, MPI, CN, DAA, CC, ML, OS, OOP, LCS, DD, DISCO, CP, CA, CB

Course Load Categories: Various (ranging from 0.5 to 1.5)

Observations - Hungarian Algorithm (Python):

Optimal Assignments: Successfully assigned courses considering individual preferences and varied course load categories. Efficiency and Scalability: Demonstrated efficiency even with increased complexity compared to Test Case 1. Handling Complexity: Managed course allocation for a larger dataset, showcasing the algorithm's scalability.

Observations - Heuristic Approach (C++):

Efficiency in Handling Large Datasets: Provided efficient assignments but lacked global optimality, similar to Test Case 1. Comparative Prioritization: Showcased varied prioritization compared to the Hungarian Algorithm, emphasizing its different approach to course assignments. Robustness: Displayed robustness in handling a more extensive dataset, highlighting its capacity to manage larger input sizes efficiently.

Conclusion of Comparative Analysis:

Hungarian Algorithm vs. Heuristic Approach:

The Hungarian Algorithm ensured optimality in course assignments, adhering strictly to preferences and constraints. The Heuristic Approach prioritized efficiency and speed but traded off optimality for faster solutions, suitable for larger datasets. Both approaches demonstrated distinct strengths, emphasizing the trade-offs between precision and computational efficiency in solving course assignment problems.

The analysis of Test Cases 1 and 2 provided insights into the performance of both algorithms under different dataset sizes and complexities, showcasing their strengths and trade-offs in optimizing the University Course Assignment System.

Crash Test Analysis Results

Hungarian Algorithm:

Extreme Workload Imbalance:

Observation: The algorithm struggled to allocate courses effectively in scenarios with imbalanced workloads. It led to overloading certain faculty categories while leaving others significantly underutilized.

Conclusion: The Hungarian Algorithm faces challenges in handling extreme workload disparities, leading to imbalanced course allocations.

Duplicate or Missing Data:

Observation: The algorithm exhibited robustness against duplicate entries but encountered issues with missing data, resulting in unpredicted course allocations.

Conclusion: Handling missing data needs further improvement to prevent unforeseen allocations in the Hungarian Algorithm.

Course-Heavy Faculty Categories:

Observation: The algorithm attempted to accommodate faculty preferences beyond the available course list, resulting in partial assignment failure or skewed allocations.

Conclusion: The algorithm needs enhancements to manage preferences exceeding available courses, ensuring feasible assignments.

Randomized Faculty-Course Allocation:

Observation: The algorithm struggled to optimize assignments when faculty preferences were randomly allocated. The resulting assignments showed a lower preference alignment.

Conclusion: While effective in preference-based allocations, the Hungarian Algorithm requires improvements in non-preference-based scenarios to maintain optimal assignments.

Heuristic Approach:

Data Overflow:

Observation: The heuristic approach exhibited commendable performance in handling datasets larger than previously tested capacities. It maintained stability but experienced increased processing times.

Conclusion: The algorithm demonstrates scalability but requires optimization to minimize computational bottlenecks in larger datasets.

Extreme Faculty-Course Preferences Disparity:

Observation: The algorithm successfully managed widely varied faculty preferences, efficiently aligning assignments even with extreme disparities.

Conclusion: The heuristic approach showed resilience in optimizing assignments despite significantly varying faculty preferences.

Limited Course-Preference Faculty:

Observation: The algorithm effectively handled scenarios with limited preference lists, ensuring reasonable assignments for faculty with constrained preferences.

Conclusion: The algorithm demonstrates adaptability in accommodating faculty members with restricted preference choices.

Unorthodox Input Data:

Observation: The heuristic approach responded well to non-standard input formats, showing adaptability to diverse data structures.

Conclusion: The algorithm exhibits robustness in processing varied input formats, suggesting a degree of flexibility and versatility.

Overall Summary

Hungarian Algorithm: Demonstrated limitations in handling workload imbalances, missing data, and non-preference-based allocations. Needs improvements in managing preferences exceeding available courses.

Heuristic Approach: Showed robustness in dealing with varying faculty preferences, limited preference lists, and non-standard data formats. Requires optimization for larger datasets to minimize processing times.

The crash test analysis provides valuable insights into the strengths and limitations of both algorithms. While the Hungarian Algorithm struggled in certain extreme scenarios, the Heuristic Approach showcased adaptability and resilience. Further refinements in both approaches can enhance their performance in handling atypical input scenarios, making them more reliable tools for course assignment optimization.

Consistency Report

Hungarian Algorithm:

1. Iterative Testing:

- Observation: The Hungarian Algorithm consistently produced similar results across multiple iterations of the same test case.

- Conclusion: The algorithm demonstrates a high level of consistency in generating assignments, indicating reproducibility in its outcomes.

2. Varying Input Parameters:

- Observation: Minor changes in input parameters, such as faculty preferences or workload capacities, occasionally led to slightly different assignments.

- Conclusion: While generally consistent, the algorithm shows sensitivity to minor variations in input parameters, affecting assignment outcomes to a limited extent.

3. Stability in Standard Scenarios:

- Observation: In standard scenarios with balanced workloads and well-defined preferences, the algorithm consistently provided stable and repeatable results.

- Conclusion: The algorithm exhibits stable behavior and high consistency in scenarios adhering to standard conditions.

Heuristic Approach:

1. Reproducibility in Diverse Scenarios:

- Observation: Across various test scenarios, the heuristic approach consistently generated similar assignments, showcasing its reproducibility.

- Conclusion: The algorithm displays a commendable level of consistency, maintaining stable results in diverse scenarios.

2. Robustness in Extreme Scenarios:

- Observation: Even under extreme workload imbalances or widely varied preferences, the heuristic approach maintained consistent assignment quality.

- Conclusion: The algorithm's consistency extends to challenging scenarios, indicating robustness in handling diverse and extreme conditions.

Overall Consistency Assessment:

- Hungarian Algorithm: Demonstrated high consistency in standard scenarios but exhibited sensitivity to minor input parameter variations.

- Heuristic Approach: Showed remarkable consistency across various scenarios, displaying stability even in extreme or non-standard conditions.

The consistency report highlights the robustness and reliability of both algorithms in generating consistent assignments across multiple iterations and varying scenarios. While the Hungarian Algorithm proved stable in standard scenarios, the heuristic approach showcased consistency in diverse and challenging conditions, indicating its reliability across a wider spectrum of inputs.