An Application of Graph Optimization-Course Project

Omkar Sathe, Om Joshi, Tanay Shah

---

1. Objective & Constraints

The research addresses the optimization challenges inherent in the University Course Assignment System, focusing on allocating courses among faculty members within a department. The faculty is categorized into three distinct groups, denoted as "x1," "x2," and "x3," each assigned 0.5 courses per semester, 1 course per semester, and 1.5 courses per semester, respectively.

Furthermore, faculty members maintain preference lists of courses, reflecting their priorities, with no prioritisation within the same category. The primary objective is formulating an assignment scheme that maximizes course assignments while adhering to faculty preferences and category-based constraints.

2. Course assignment algorithm

An unweighted bipartite graph is first generated where edges map each professor to all the courses including CDCs and electives given in their preference. The problem statement now becomes to find the Maximum Cardinality Bipartite Matching (MCBM). Our implementation of the course assignment problem uses the max flow algorithm for bipartite graph matching.

This bipartite matching problem is converted to a network flow problem. All the edges of the bipartite graph are made into directed edges, and given maximum flow capacity as 1. Additional sink and source nodes are added. The source node is attached to all the professors and the sink node is attached to all courses. The edges originating from the source node have been assigned flow capacity as 0.5, 1 or 1.5 depending on the type of professor (x1, x2, x3). Edges going to the sink node have a flow capacity of 1.

Thus, a flow network is generated and a max flow algorithm can be applied. Here, we have used the Ford Fulkerson algorithm for finding the max flow.

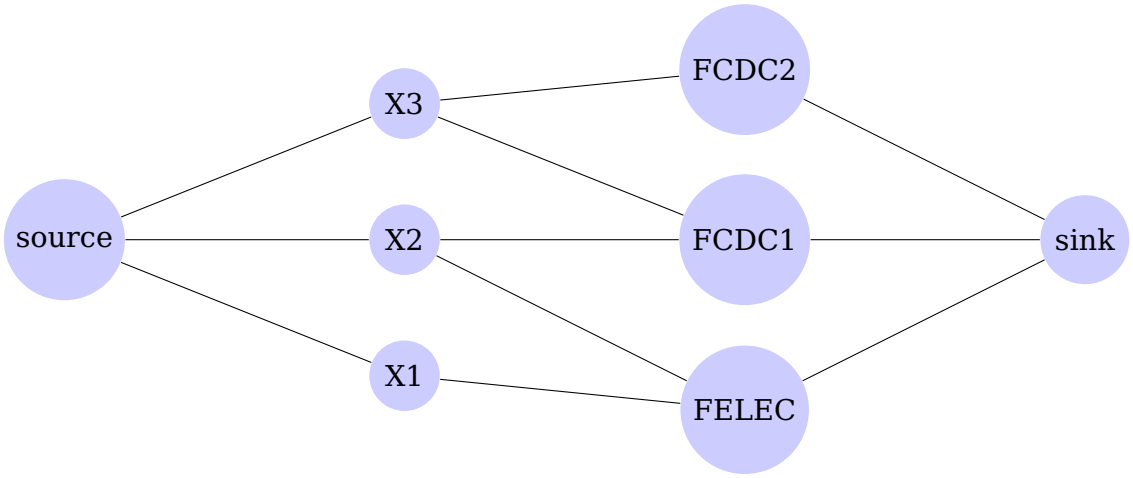
3. Ford-Fulkerson Network Flow Algorithm

For our implementation of Ford-Fulkerson, we have used depth-first search (DFS) for finding augmenting paths in a network flow graph.

The DFS algorithm navigates the flow graph to identify augmenting paths. This process, represented by the DFS method, mimics the decision-making of the university as it traverses their preference lists and adjusts their course assignments. The iterative augmentation ensures that the algorithm explores different paths, capturing the fluidity of course allocations

Course Assignment Strategies: The addEdge method encapsulates the logic for incorporating directed edges into the graph. This step reflects the translation of faculty preferences and constraints into the flow graph. Residual edges enable the algorithm to adapt and explore alternative paths during subsequent iterations, mirroring the flexibility needed for dynamic course assignments.

Execution and Result Analysis: The solve method orchestrates the overall execution, mirroring the academic semester. The repeated invocation of the DFS algorithm symbolizes the iterative nature of course allocation. The resulting maximum flow embodies the optimal allocation of courses, aligning with faculty preferences and category-specific constraints.



Example course assignment

4. References

Ford, L. R., Fulkerson, D. R. (1956). Maximal flow through a network. Canadian Journal of Mathematics, 8, 399–404. https://doi.org/10.4153/cjm-1956-045-5

Mount, D. (2017, November 2). CMSC 451: Lecture 15 Network Flows: The Ford-Fulkerson Algorithm. www.cs.umd.edu. Retrieved November 27, 2023, from https://www.cs.umd.edu/class/fall2017/cmsc451-0101/Lects/lect15-flow-ford-fulk.pdf