**Research Narrative for Practical Optimization of Course Schedules with Linear Programming:**

**Linear Constraints and Data Wrangling**

**Background and Significance**

Twice each year, department chairs and ADAs at UWEC must plan the course schedule for the following semester. This is a challenging task that requires consideration of instructor teaching preferences and schedule constraints. Linear programming is a mathematical modeling approach which represents a problem using a function to be maximized (the objective function), subject to a set of constraints (equations or inequalities). It is frequently used for scheduling problems [1]. For example, suppose that two courses must be taught by two instructors, who rate their happiness with each course as follows:

|  |  |  |
| --- | --- | --- |
|  | Instructor A | Instructor B |
| Course 1 | 4 | 2 |
| Course 2 | 1 | 3 |

(Here, higher numbers indicate greater happiness.) The goal of maximizing total happiness can be represented by the objective function , where indicates that instructor A will teach class 1, and indicates that they will not. Simply maximizing this objective function would lead to both instructors teaching both classes. We can represent the constraint that each instructor should teach one class using the equations and . Thus, the goal is to maximize the objective function, but to allow only the possibilities that satisfy the constraints.

When the variables in a linear programming problem can take continuous values, the problem can be solved using the simplex method [2]. However, in course scheduling, the variables (such as ) are integers; an instructor can’t teach half a class. The optimal integer solution may be quite different from the optimal continuous solution [3], so the simplex method no longer guarantees an optimal solution. Instead, the branch-and-bound method can be used [4]. In this approach, when the optimal simplex solution involves a non-integer (say, ), we split the problem into two branches by adding a constraint: either or . Each branch is then solved using the simplex method, and the process iterates until the optimal solution is found.

**Previous Work**

In the 2022-23 ORSP-funded project “Optimizing Course Schedules with Linear Programming,” Annabelle Piotrowski wrote Python code to model and assign course schedules using linear programming in test examples of up to 15 sections of 5 courses among up to 5 instructors. Annabelle presented her research at NCUR. I used the current version of her code in July 2023 to assist with devising the Spring 2024 course schedule. In July 2023, I submitted a proposal for the project “Practical Optimization of Course Schedules with Linear Programming,” with Annabelle as the research student. That proposal was funded.

**Project Aim**

The current proposal is to add a second research student, Katherine McCallum, to the project, to enhance our ability to refine our existing code to model course scheduling as a linear programming problem, to make it effective as a practical aid to course scheduling in the UWEC math department.

**Methodology**

Katherine’s work will focus on improving two main aspects of our code. First, we will develop a system for inputting constraints about instructor-time pairs or instructor-course pairs. Our current code allows an instructor to specify that they do not prefer a certain time; we also wish to allow instructors to specify that they *cannot* teach at a particular time (and similarly for courses). This can be done by requiring for a given instructor *,* course *c,* and time *t*; the more interesting challenge is how to input which combinations are not allowed. One possibility is to use a spreadsheet in which each instructor is associated with a Python list of infeasible times, and use nested *for* loops to iterate through instructors and times. This will be a valuable complement to the work Annabelle will be doing to expand the variety of types of instructor preferences that our model can handle.

Second, we will improve data wrangling both within our existing code and to prepare the input about instructor preferences. Our existing code employs dataframes with multiple, nested levels (for example, course sections are nested within instructors, because one instructor may teach multiple sections). Combining dataframes with different numbers of levels using the *merge* function is deprecated in Python, so this code should be refined to ensure that our model continues to be usable in future versions of Python. Based on Katherine’s experience in CS 260, she has suggested solving this challenge by making use of the robust dataframe-joining capabilities of SQL, which can be integrated with Jupyter Notebooks [5], the platform we are currently using for our Python code.

Wrangling the input data about instructor preferences will also be crucial to making our model a practical aid to course scheduling. Currently, possible meeting times (for 50- and 110-minute courses) are given indices 0 to 84. However, manually translating stated preferences such as “I don’t like to teach at 8 am” into preference values of 0 for index 0 (8-8:50 on Monday), 9 (8-8:50 on Tuesday), etc. is a barrier to scaling up the use of our code. Katherine will develop a Google Survey to solicit instructor preferences and constraints, and develop Python code to arrange these data into a format suitable for our linear programming model.

**Dissemination**

Katherine will join the other student on this project, Annabelle Piotrowski, in presenting their results at the Department of Mathematics Retreat and the UWEC Celebration of Excellence in Research and Creative Activity. The students’ work will be used as a baseline for constructing instructor schedules in the math department in 2024-2025.

**References**

[1] Kolman B, Beck RE. 1995. *Elementary Linear Programming with Applications, 2nd ed*. San Diego: Academic Press.

[2] Nelder JA, Mead R. 1965. “A simplex method for function minimization.” *The Computer Journal*. 7(4):308-13.

[3] Haseen S, Sadia S, Bari A, Ali Q. 2014. “Integer programming: NAZ cut and A-T cut.” *International Journal of Engineering Science and Technology*. 06:128-137.

[4] Brucker P, Jurisch B, Sievers B. 1994. “A branch and bound algorithm for the job-shop scheduling problem.” *Discrete Applied Mathematics*. 49(1-3):107-27.

[5] Radecic D. 2020. “Here’s how to run SQL in Jupyter Notebooks.” Accessed 15 September 2023. <https://towardsdatascience.com/heres-how-to-run-sql-in-jupyter-notebooks-f26eb90f3259>