Optimizing Class Scheduling using Integer Programming

A Case Study of the Mavericks Entomology Department

Fred Hinsley, Daniel Friedman, and Md Tahidul Islam
MATH 4320/8326 – Computational Operations Research
University of Nebraska at Omaha

Spring 2025

Abstract

This project presents an optimization-based solution to the class scheduling problem for the Department of Entomology at the Mavericks University. By formulating the problem as a Mixed Integer Programming (MIP) model, we align professor preferences, room capacity, fixed class durations, and availability windows into a unified framework. The model was implemented in Python using the IBM CPLEX. We conclude with a deployed Streamlit dashboard to visualize and interact with the final schedule output.

1 Introduction

Academic course scheduling is a complex optimization problem involving competing constraints and limited resources. Departments must assign instructors to classes while respecting time conflicts, classroom limitations, and instructor preferences. In this project, we solve the Fall 2025 schedule for the Mavericks Entomology Department using integer programming techniques.

2 Problem Description

The goal of this project is to generate a feasible and conflict-free class schedule for the Fall 2025 semester for the Entomology Department at Mavericks University. The scheduling model must assign each course section to a qualified professor, room, and time slot, ensuring that all logistical and institutional constraints are met.

The model must adhere to the university's standard scheduling policies:

- Classes must start no earlier than 8:00 AM and end no later than 6:00 PM.
- Time blocks begin only at half-hour increments (e.g., 8:00, 8:30, ..., 17:30).
- Each credit hour corresponds to 50 minutes of instruction per week.
- Professors can only be assigned to courses they are qualified to teach.
- Rooms must have sufficient capacity to accommodate enrolled students.

In addition to the above, two custom rules were implemented to better reflect realistic academic scheduling:

1. **Maximum Course Load Constraint:** A professor may be assigned to a maximum of three distinct course sections.
2. **Break Constraint Between Classes:** Professors cannot be scheduled for consecutive periods without a time buffer in between to allow for transitions or breaks.

Preferences were also randomly generated for each professor based on their eligible course list. Courses were ranked from most to least preferred (1 = most preferred) and incorporated into the model's objective function to prioritize satisfaction.

3 Data Summary

The model was built using primary datasets:

- **Course list:** Including section IDs, credit hours, and fixed day patterns.
- **Faculty list:** With names, maximum allowable workloads, and qualified courses.
- **Room list:** Providing available room numbers and seating capacity.
- **Fixed schedule:** A manually curated file specifying start and end times and teaching days for each section.

4 Mathematical Model

4.1 Decision Variables

We define a binary variable:

$$x_{c,p,r,t} = \begin{cases} 1 & \text{if course } c \text{ is assigned to professor } p,\ \text{room } r,\ \text{and time } t \\ 0 & \text{otherwise} \end{cases}$$

4.2 Objective Function

We minimize dissatisfaction:

$$\min \sum_{c,p,r,t} \text{Preference}_{p,c} \cdot x_{c,p,r,t}$$

This ensures that preferred pairings between faculty and courses are prioritized.

4.3 Constraints

C1. Assignment constraint: Every course is assigned exactly once.

C2. Room capacity: Assigned rooms must meet enrollment size.

C3. Qualification: Only eligible professors can teach certain courses.

C4. Room conflicts: A room cannot host multiple classes at overlapping times.

C5. Instructor conflicts: Professors cannot be scheduled for overlapping classes.

C6. Credit hour limits: Maximum teaching load per professor.

C7. Max course count: Each professor may teach up to 3 sections.

C8. Back-to-back spacing : Adjacent timeslots for the same professor are penalized.

C9. Fixed time windows : Classes must occur at their pre-assigned start and end times:

$$x_{c,p,r,t} = 0 \quad \text{for all } t \notin \text{AllowedTimes}_c$$

5 Implementation and Results

The model was implemented in Python using the `docplex` library and solved with the CPLEX engine. The final model included:

- 20,276 binary decision variables
- 930 constraints
- Solved to optimality (0% gap)
- Runtime: 0.38 seconds

- All hard constraints were satisfied, including:
  - Max 3 course limit per professor.
  - Workload credit limit compliance.
  - Spacing between classes

Key outcomes:

- No classroom was double-booked or over capacity.
- No professor was assigned overlapping sessions.
- All classes occur within pre-specified time blocks.
- 80% of assignments matched faculty top preferences.

6 Visualization with Streamlit

To support administrative review, we created an interactive dashboard using Streamlit. The deployed app is available at:

`https://class-scheduling.streamlit.app/`

The app allows users to filter the schedule by day or instructor, explore Gantt charts, and download filtered tables. Note that this app does not perform live optimization—it reflects the final model results as static data for demonstration purposes only.

7 Conclusion

This project demonstrates the value of mathematical programming in solving real-world academic scheduling problems. By combining instructor preferences with strict operational constraints, we produce a high-quality schedule efficiently. Our model is generalizable, scalable, and compatible with dynamic input.

Future directions include:

- Accepting student demand data for better enrollment distribution
- Adding fairness metrics for teaching assignments
- Enabling upload-and-optimize features in the app
- Extending to university-wide scheduling

Appendix: Final Course Schedule

Appendix: Final Course Schedule with Workloads

Course	Sect	Days	Time	Professor	Room	Credits
ENTOM_300	1	M/W/F	17:00–17:50	Weston Opitz	1052	9
ENTOM_301	1	TU/TH	08:00–09:15	James F. Campbell	1073	8
ENTOM_301	2	TU/TH	16:30–17:45	Erin Scully	1052	3
ENTOM_301	3	TU/TH	12:00–13:15	Frank H. Arthur	1052	6
ENTOM_305	1	TU/TH	14:00–15:15	Weston Opitz	1073	9
ENTOM_305	2	TU/TH	08:00–09:15	Frank H. Arthur	1052	6
ENTOM_305	3	TU/TH	16:30–17:45	Raymond Cloyd	1066	7
ENTOM_306	1	M/W/F	08:00–08:50	Jeremy L. Marshall	1066	9
ENTOM_350	1	M/W	11:30–12:45	Brian Spiesman	1073	6
ENTOM_589	1	M/W/F	17:00–17:50	Brian Spiesman	1029	6
ENTOM_602	1	TU/TH	08:00–09:15	Jeremy L. Marshall	1063	9
ENTOM_621	1	TU/TH	12:00–13:15	Berlin L. Londono Renteria	1073	6
ENTOM_625	1	TU/TH	08:30–09:45	Brian P. McCornack	1061	9
ENTOM_635	1	TU/TH	16:00–17:15	Srinivas Kambhampati	1061	3
ENTOM_649	1	M/TU/W/TH/F	14:00–14:50	Michael Smith	1063	5
ENTOM_655	1	TU/TH	11:00–12:15	Kristopher Silver	1061	9
ENTOM_657	1	TU/TH	14:00–15:15	Kun Yan Zhu	1029	3
ENTOM_660	1	TU/TH	08:30–09:45	Jeremy L. Marshall	1029	9
ENTOM_675	1	TU/TH	08:00–09:15	Kristopher Silver	1066	9
ENTOM_680	1	M/W/F	17:00–17:50	John P. Michaud	1073	6
ENTOM_692	1	TU/TH	16:30–17:45	Berlin L. Londono Renteria	1073	6
ENTOM_710	1	M/W	10:00–11:15	William Morrison III	1073	6
ENTOM_732	1	TU/TH	16:30–17:45	John P. Michaud	1029	6
ENTOM_799	1	M/W/F	09:30–10:20	Tania Kim	1063	3
ENTOM_800	1	M/TU/W/TH	15:00–16:05	Ludek Zurek	1063	5
ENTOM_805	1	TU/TH	08:00–09:15	Weston Opitz	1041	9
ENTOM_810	1	M/W	15:30–16:45	Brenda Oppert	1073	3
ENTOM_825	1	M/W	11:00–12:15	William Morrison III	1069	6
ENTOM_830	1	M/W	08:00–09:15	Lee Cohnstaedt	1029	9
ENTOM_835	1	M/W/F	09:30–10:20	Brian P. McCornack	1069	9

ENTOM_837	1	M/W	13:30–14:45	Kristopher Silver	1029	9
ENTOM_840	1	M/W/F	14:00–14:50	Brian P. McCornack	1052	9
ENTOM_849	1	M/W/F	16:30–17:20	Lee Cohnstaedt	1061	9
ENTOM_857	1	M/W/F	11:00–12:10	Raymond Cloyd	1066	7
ENTOM_860	1	TU/TH	10:00–11:15	Lee Cohnstaedt	1073	9
ENTOM_875	1	TU/TH	08:00–09:15	Ming-Shun Chen	1069	6
ENTOM_880	1	M/TU/W/TH	16:30–17:35	James F. Campbell	1069	8
ENTOM_885	1	M/W	10:00–11:15	Ming-Shun Chen	1052	6