Final Report

July 29, 2024

**CS/ECE/ISyE 524 — Introduction to Optimization — Summer 2024An university course timetabling problem**

**Student 1 (email address), Student 2 (email address), Student 3 (email address)**

**Table of Contents**

1. Introduction

2. Mathematical Model

3. Solution

4. Results and Discussion

5. Optional Subsection

6. Conclusion

1. **Introduction**

Timetabling problems are a specific type of scheduling problem that manages assignments of events into timeslots. These assignments could be subject to a number of hard and soft requirements, the goal is to find a valid assignment that satisfies all hard-requirements while satisfying as much soft-requirements as possible [1].

There are many variations and applications of this problem across different fields, such as airport slot-scheduling, sport scheduling, logistical scheduling, and university timetabling. Of these variations, the field we feel that is most applicable to us is university timetabling, which could be divided into examination timetabling and course timetabling problems [2], we chose to address the University Course Timetabling Problem (UCTP).

Within the typical timetabling problem an education institution might face, we further narrow our focus on the allocation of instructors of courses into fixed timeslot within a single week.

There are a still lot to consider when scheduling courses in a single week, instructors might only be available at certain times, courses could be mandated to not be scheduled at certain dates, there might be instructor teaching multiple subjects, multiple instructors could be available for a single course, classes of the same year might have different required courses.

To avoid too much complexity, the scenario we consider is as follows: There are several classes, each with one or more courses that needs a fixed number of timeslots each week, with a single corresponding instructor, each weekday has a fixed number of timeslots available for all the classes.

In addition to guaranteeing the schedule has no conflict, we limit the added hard requirements to two types, the case that a certain instructor might be unavailable for certain timeslots, or whenever a certain course must be at a certain timeslot.

For the soft requirement, we reckon modeling the preference of either instructor or student might be the most suitable. Drawing from our own experience, we found that having multiple repeated courses in a single day is generally disliked, so the soft requirement we choose is to minimizing the number of repeated courses in a single weekday.

**1.1 Handcrafted Data**

To confirm our model design works as intended, we will start from a simple handcrafted data with small number of classes and instructor. For simplicity, we start with two classes, nine instructors, and seven different courses, with four timeslots per weekday.

The following table shows the handcrafted course data we created for the proof of concept.

|  |  |  |  |
| --- | --- | --- | --- |
| **Instructor** | **Course** | **# of timeslot for Class 1** | **# of timeslot for Class 2** |
| Mr. Cheese | English | 1 | 1 |
| Mrs. Insulin | Biology | 3 | 3 |
| Mr. Map | History-Geography | 2 | 2 |
| Mr. Effofecks | Mathematics | 0 | 4 |
| Mrs. Derivative | Mathematics | 4 | 0 |
| Mrs. Electron | Physics | 3 | 3 |
| Mr. Wise | Philosophy | 1 | 1 |
| Mr. Muscle | Sport | 1 | 0 |
| Mrs. Biceps | Sport | 0 | 1 |

In this iteration of crafted data, we keep the courses and number of timeslots needed the same for the two classes, some of the instructor teaches both classes, the others would only manage one of the classes.

For the additional hard requirements, we marked some instructors as unavailable for some time slots and fixed the sport course to the third timeslot on Thursday.

The following tables shows instructors that is unavailable at certain times.

|  |  |  |
| --- | --- | --- |
| **Instructor** | **Weekday** | **timeslot** |
| Mr. Effofecks | Monday | 1 |
| Mr. Effofecks | Monday | 2 |
| Mrs. Insulin | Wednesday | 1 |
| Mrs. Insulin | Wednesday | 2 |
| Mrs. Insulin | Wednesday | 3 |
| Mrs. Insulin | Wednesday | 4 |

The following table shows courses that must occur at certain time.

|  |  |  |
| --- | --- | --- |
| **Course** | **Weekday** | **timeslot** |
| Sports | Thursday | 3 |

The code we used to represent this handcrafted data is as follows:

一張含有 文字, 螢幕擷取畫面, 數字, 軟體 的圖片

自動產生的描述

1. **Mathematical Model**

To find a valid schedule, we need to first ensure that every class has all the required courses allocated with no extra, this can be done by adding constraints for all courses of a certain class, that the total number scheduled is equal to the number planned for the week.

The next problem to tackle is conflicts in the schedule, whether it is an instructor assigned to multiple courses at a single time slot, or a class that has multiple courses scheduled in a single time slot. We limit the total number of courses assigned to any instructor at a single time slot to be less than or equal to one, and the total number of courses scheduled for any class at a single time slot to be less than or equal to one.

For the task of setting the hard requirements, we added upper and lower bounds for each course that is separate for every time slot. We can achieve the desired effect by setting the upper bounds to zero on timeslots where a certain instructor is unavailable, and setting the lower bounds to one when a course is mandatory for a certain timeslot.

For the soft requirement, the goal is to minimize repeated courses on a single weekday, we model this by calculating the total count of the same course over a single day and subtract this by 1, and clamp the value to be at least 0, this is so having a single course in a day is not penalized and having no course in a day is not rewarded. The sum of this value across all weekdays is considered the penalty score for a schedule, which is the target variable to be minimized.

* + 1. **UCTP Model**

**Data***N* – Set of timeslots per day *M* – Set of weekdays   
*K* – Set of instructors-course pairs  
*B* – Set of classes  
*ckb* – Number of courses by instructor for class   
*uijkb* – Upper bound of scheduled courses by instructor at timeslot of weekday for class   
  
*lijkb* – Lower bound of scheduled courses by instructor at timeslot of weekday for class  **Variables***xijkb* – Whether course by instructor for class is scheduled for timeslot of weekday  **Model**一張含有 文字, 字型, 螢幕擷取畫面, 筆跡 的圖片

自動產生的描述Constraint (1) restricts the number of courses can be scheduled for every instructor at each timeslot to be at most 1. Constraint (2) enforces the required courses for each class is assigned, and no unnecessary course is scheduled Constraint (3) and (4) ensures no conflict occurs for both instructors and the classes. Constraint (5) and (6) are the previously mentioned upper and lower bounds to implement the additional hard requirements.

1. **Model Implementation**

Before we could set our constraints and objectives, the unavailable instructor and fixed courses data must be transformed into suitable data structures. 一張含有 文字, 螢幕擷取畫面, 字型, 數字 的圖片

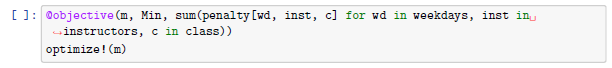
自動產生的描述With the data ready, we could now begin setting up the model itself.

一張含有 文字, 螢幕擷取畫面, 字型, 數字 的圖片

自動產生的描述

For the objective function, there are extra complications, since we want to clamp the penalty of having repeated courses to greater or equal to zero, we wanted to use a max function, but this cannot directly be done in a linear formulation. We would need to define an auxiliary binary variable to formulate a constraint for the penalty function to ensure that any negative value is set to zero.[3]  
一張含有 文字, 螢幕擷取畫面, 字型, 數字 的圖片

自動產生的描述  
We can now add the objective to the model and run the optimizer.



The optimizer finishes after 34 iterations, we output the generated schedule using PrettyTables. 一張含有 文字, 螢幕擷取畫面, 字型, 數字 的圖片

自動產生的描述

一張含有 文字, 螢幕擷取畫面, 數字, 字型 的圖片

自動產生的描述

Observing the generated schedule, we can confirm our model creates schedule that minimize repeated courses while maintaining all the hard requirements.

1. **Results and Discussion**

During our own discussion, we find that our current model actually supports slightly more complex scenarios then we designed it for. Courses for different classes does not necessarily needs to be the same, different classes can also have different time slot that is available, this can be achieved by setting the upper bound of time slots not available to zero.

While there are some surprises to our model’s capabilities, we could still find many limitations. Currently we fixed every instructor to a single course, and we treat this instructor-course pair as a whole, this means any instructor that teaches multiple courses would generate multiple pairs that our model treat as completely different courses, thus would not find scheduling two courses at the same time as a conflict.

1. **Conclusions**
2. **References**

[1] Tomáš Müller, “*Constraint-based Timetabling”*, Ph.D. Thesis, Faculty of Mathematics and Physics, Charles University in Prague, Prague, 2005.

[2] Nancy Maribel Arratia-Martinez, Cristina Maya-Padron, and Paulina A. Avila-Torres, “University Course Timetabling Problem with Professor Assignment” Mathematical Problems in Engineering. [Online]. Available: <https://onlinelibrary.wiley.com/doi/10.1155/2021/6617177>

[3] user477602, linear programming set a variable the max between two another variables, URL (version: 2017-09-27): https://math.stackexchange.com/q/2447498