"The Examination Timetabling Problem Project report "

Profs. R. Rizzi, R. Zanotti

Discrete Optimization and Decision Making - 2022/2023

Mahsa Sedaghatinia / VR481427

October 2023

Contents

1 Project definition

Let us consider a set E of exams, to be scheduled during an examination period at the end of the semester, and a set S of students. Each student is enrolled in a non-empty subset of exams. The examination period is divided into T ordered time-slots. Given two exams $e_1, e_2 \in E$, let n_{e_1,e_2} be the number of students enrolled in both. Two exams $e_1, e_2 \in E$ are called conflicting if they have at least one student enrolled in both, i.e., if $n_{e_1,e_2} > 0$. Rules and regulations impose that conflicting exams cannot take place in the same time-slot. Moreover, to promote the creation of timetables more sustainable for the students, a penalty is assigned for each pair of conflicting exams scheduled up to a distance of 5 time-slots. More precisely, given two exams $e_1, e_2 \in E$ scheduled at distance i of time-slots, with $1 \leq i \leq 5$, the relative penalty is $2^(5 - i) \cdot \frac{n_{e_1,e_2}}{|S|}$.

The ETP aims at assigning exams to time-slots ensuring that:

- each exam is scheduled exactly once during the examination period;
- two conflicting exams are not scheduled in the same time-slot;
- the total penalty resulting from the created timetable is minimized.

1.1 data

- exams: $E = \{e_1, e_2, \ldots, e_n\}$
- students: $S = \{s_1, s_2, \ldots, s_m\}$
- time slots: $T_j = \{t_1 \leq t_2 \leq \ldots \leq t_T\}$ (ordered)
- n_{e_1,e_2} = number of common students for e_1 and e_2. If $n_{e_1,e_2} > 0$, then e_1 and e_2 are conflicting.

1.2 constraints

- each exam is scheduled exactly once during the examination period;
- two conflicting exams are not scheduled in the same time-slot;

1.3 objective function

The total penalty resulting from the created timetable is minimized.

$$\min \sum_{e_1,e_2 \in \text{conflicting\_exams}} p_{e_1,e_2}$$

2 Project Mathematical Modeling

2.1 initializing data and parameters

The instances raw data files were used to extract the data and then the extracted data was stored.

- **exams=[ ]**, List of exams
- **time_slots**, List of time slots from 1 to T (ordered)
- **enrollment=dict()**, dictionary for the enrollments. The students are the keys and a list of exams belongs to each key.
- **students**, keys of the enrollment dictionary (students=enrollment.keys())
- **conflicting_exams = {}**, dictionary for the conflicting exams. For every pair of conflicting exams we indicated n_{e_1,e_2}, the number of conflicts. We store only one order of a pair. For example if exam e_1 and exam e_2 are conflicting, then (e_1, e_2) or (e_2, e_1) is a key in the dictionary, but not both of them

2.2 modeling

$x_{t,e} \in \{0, 1\}$, binary variable declaring if exam e is scheduled in time slot t

$$x = \begin{cases} 1, & \text{if exam e is scheduled in t} \\ 0, & \text{otherwise} \end{cases}$$

using the x variable the first 2 constraints can be achieved by the following expressions:

1. Each exam is scheduled exactly once during the examination period.

$$\sum_{j=1}^{T} x_{t_j,e} = 1, \forall e \in E$$

2. Two conflicting exams are not scheduled in the same time-slot

$$x_{t,e_1} + x_{t,e_2} \leq 1, \forall t \in T_j, \forall (e_1, e_2) \in \text{conflicting\_exams}$$

Two variables were defined to state if 2 conflicting exams are scheduled within a given (i) distance. In this part two different scenarios had been considered for two conflicting exams (exam 1 and exam 2): first, if exam 1 is scheduled at time slot t and exam 2 is scheduled at time slot t + i (later than exam 1). second scenario is the opposite situation where exam 1 is scheduled later than exam 2. for expressing this scenarios we have 2 variables:

1. $a[distance, time\_slot, exam_1, exam_2] \in \{0, 1\}$, binary variable for scheduling $exam_1$ before $exam_2$

2. $b[distance, time\_slot, exam_1, exam_2] \in \{0, 1\}$, binary variable for scheduling $exam_2$ before $exam_1$

It is clear that the values of a and b can not be equal to 1 simultaneously, for scheduling each pair conflicting exams with respect to this condition the two following constraint were introduced:

1. $a[i, t, exam_1, exam_2] >= x[t, exam_1] + x[t+i, exam_2] - 1$ exam 2 is scheduled after exam 1

2. $b[i, t, exam_1, exam_2] >= x[t+i, exam_1] + x[t, exam2] - 1$ exam 1 is scheduled after exam 2

introducing and using these two new variables is handy since it will make sure that for any two conflicting exams, they will be scheduled with a distance and a penalty will be calculated for each conflicting pair. We later define the integer variable p as the sum between a and b. With this variable we can express when two conflicting exams are scheduled at a given distance taking in consideration a given time slot:

1. $p[i, t, exam_1, exam_2] = sum(a[i, t, e1, e2] + b[i, t, e1, e2])$ for t in range(1, $len(time_slots)$ - i + 1 exam 2 is scheduled after exam 1

Finally the objective function was defined based on p, which is given by the following formulation:
sum(p[i, exam1, exam2] * (2**(5-i)) * conflicting_exams.get((exam1, exam2), 0) / len(students)

3 Results

The results of this optimization model are assessed by evaluating the "Penalty" value achieved for a given instance, the lower the penalty is the better the exams are scheduled. Below there is a table of achieved penalties:

instances	penalty value
Test	3.375
instance 01	162.585
instance 02	53.359

Table 1: Penalty values of the instances