DODM23 Project

Examination Timetabling Problem (ETP)

Laura Zanetti

1. Problem Description

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 e1, e2 ∈ E, let $n_{e1\,e2}$ be the number of students enrolled in both. Two exams e1, e2 ∈ E are called conflicting if they have at least one student enrolled in both, i.e., if $n_{e1\,e2} > 0$.

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 e1, e2 ∈ E scheduled at distance i of time-slots, with 1 ≤ i ≤ 5, the relative penalty is $2^{|5-i|} \cdot n_{ei\,ej}/|S|$.

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.

Assumption: in each time-slot there are enough resources (number of rooms, capacities) to accommodate all the exams and all the enrolled students

2. Problem Formulation

Let us define now the ETP sets and parameters used in the formulation:

- E = {1, 2, ..., N} is the set E of exams, with examID.
- T = {1, 2, …, Tmax} is the set T of time-slots.
- S = {1, …, M} the set of students, with studentID.
- n_{ij} (Also, as $n_{ei\,ej}$) = number of students enrolled in both exams i and j.

$$(i \; and \; j \; are \; conflicting \; if \; n_{ij} > 0)$$

- P = a vector containing the distances from 1 to 5.
- p_d=2^{5-d} , the exponential contributes to the penalty = [16, 8, 4, 2, 1], with d ∈ P.

The aim is to know in which timeslot to schedule every exam, taking into account some hard and soft constraints. Let's consider then:

Decision variables:

- A binary variable x_{it} which allow us to establish if exam i is assigned or not to timeslot t.

$$x_{it} = \begin{cases} 1\ ,\ if\ an\ exam\ i\ is\ assigned\ to\ timeslot\ t \\ 0\ ,\ otherwise \end{cases}$$

- An auxiliary variable which allows us to take into consideration the penalty we encounter with the pair of exams i and j:

y_{ij} = penalty incurred over the pair i and j.

Constraints:

- Each exam is schedule exactly once during the examination period (to every exam is associated only one timeslot):

$$\sum_{t=1}^{T_{max}} x_{it} = 1 \qquad \forall i \in E$$

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

$$(x_{it} + x_{jt}) \leq 1 \qquad \forall\ i, j \in E, \forall\ t \in T,\ n_{ij} > 0$$

- Through the following constrain we define variable y_{ij} and connect it with variable x_{it}:

$$y_{ij} \geq p_d * x_{it} + p_d * x_{jt+d} - p_d$$
$$y_{ij} \geq p_d * x_{it+d} + p_d * x_{jt} - p_d$$

$$\forall\ i, j \in E, \forall\ t \in T, \forall d \in P, i < j, t+d <= tmax, n_{ij} > 0$$

Objective Function:

We want to minimize the penalty over all the couple of exams i and j, considering $n_{ij} > 0$:

$$\text{Min} \quad \sum_i^E \sum_j^E y_{ij}$$

3. Implementation with Gurobi

After writing the ILP formulation for the Examination Timetabling Problem the following step was to implement it in Gurobi using Python. After importing the various needed libraries, the instances files were read in order to obtain the length of the examination period (instanceXX.slo), the number of enrolled students per exam (instanceXX.exm) and the exams in which each student is enrolled (instanceXX.stu). Once asserted the consistency of the input data for each instance, the vectors and matrices of known values where created, such as the exponential contributes to the penalty p[d] (distance d between 1 and 5) and the conflict matrix n[i, j]. Also, the lists of exams and timeslot were created.
The environment for the model was then set up along with some parameters decided according to the problem and resolution desired:

- env.setParam("Threads", 3) →Controls the number of threads to apply to parallel algorithms.
- env.setParam("Presolve", 1) → Controls the presolve level. conservative (1).
- env.setParam("MIPGap", 1e-4)
- env.setParam('Method', 0) → Algorithm used to solve the initial root relaxation of a MIP model. 0=primal simplex.
- env.setParam("TimeLimit", 600) → 10 minutes time limit (sometimes increased to 1200s).
- env.setParam("PreSparsify", 1) → to reduce the number of non-zeros in the presolved model (reducing the memory used) and avoid this way going out of memory (especially happening with large instances).

Following, the model is initialized and the variables are added to it as well as the constraints and the objective function. The model is then optimized and run.

Once the model finds a feasible solution the ETP_ProblemXX.sol file is created containing the found solution, the check for feasibility and the penalty.

At last, the model and the environment were disposed of.

4. Solving the 11 benchmark instances

The 11 instances + the test were all run once at a time with the following results:

- **Test** → an optimal solution was found with objective function value of 13.
- **instance01** → a feasible (not optimal) solution was found within the time limit of 10 minutes with a value for the objective of 5133.
- Instance02 → doesn't return a feasible solution in the time limit considered (also tried by doubling it)

- Instance03 → doesn't return a feasible solution in the time limit considered (also tried by doubling it)
- Instance04 → doesn't return a feasible solution in the time limit considered (also tried by doubling it).
- Instance05 → doesn't return a feasible solution in the time limit considered (also tried by doubling it).
- Instance06 → it seems the instance is too big and when the model is run it is not optimized (in the console doesn't appear anything after the parameters set)
- **Instance07** → a feasible solution was found within the time limit of 10 minutes with a value for the objective of 4303.
- **Instance08** → a feasible solution was found within the time limit of 10 minutes with a value for the objective of 5879.
- Instance09 → doesn't return a feasible solution in the time limit considered.
- Instance10 → it seems the instance is too big and when the model is run it is not optimized (in the console doesn't appear anything).
- Instance11 → it seems the instance is too big and when the model is run it is not optimized (in the console doesn't appear anything).

In the end feasible solutions were found just for the Test, Instance01, Instance07 and Instance08.

For every Instance a log (.log) file was created. A solution (.sol) file was also created, but just in the case a feasible solution was found.

Advanced Model

In the following sections some equity measure and additional constraints were elaborated (tried to):

5. Equity Measures:

- **Number of students with back-to-back exams**: This measure is the number of students who have two conflicting exams scheduled in consecutive time-slots. A timetable with a low number of students with back-to-back exams is more equitable for students, as it means that they will have more time to rest between exams.
  The objective function would become:

$$\min \sum_i^E \sum_j^E (x_{it} + x_{jt+1}) \qquad \forall t \in T,$$
$$n_{ij} > 0$$

6. Additional restrictions:

- At most 3 consecutive time slots can have conflicting exams:

$$\sum_{i,j,k \in E} (x_{it} + x_{jt+1} + x_{kt+2}) \leq 3 \qquad \forall t \in T,$$
$$i,j,k\ conflicting$$

- Change the constraints that impose that no conflicting exams can be scheduled in the same time slot. Instead, impose that at most 3 conflicting pairs can be scheduled in the same time slot:

$$\sum_i^E \sum_j^E \sum_l^E (x_{it} + x_{jt} + x_{kt}) \leq 3 \quad \forall t \in T,\ i,j,k\ conflicting$$