Optimising Scheduling of Hybrid Learning using Mixed Integer Programming

Matthew Davison[1], Ahmed Kheiri[2], Konstantinos Zografos[2]

[1] STOR-i Centre for Doctoral Training, Lancaster University, Lancaster, UK
m.davison2@lancaster.ac.uk
[2] Department of Management Science, Lancaster University, Lancaster, UK
{a.kheiri,k.zografos}@lancaster.ac.uk

Keywords: University Timetabling · Hybrid Learning · Mixed Integer Programming

1 Introduction

The COVID-19 pandemic artificially reduced the already limited capacity of physical spaces at universities. It forced a greater use of *hybrid learning*, which is a mode of teaching that combines online and in-person elements. Previous studies dealt with limited capacity by controlling the quantity and flow of students on campus [5] or on the areas surrounding campus [1]. Other studies investigated what policy changes allow a better use of resources [2].

Online classes are one way to reduce demand for physical space. Universities in the future will likely continue to offer a mix of online and in-person classes beyond the pandemic [3]. Since students and staff typically prefer to attend classes in-person, this motivates the need to limit the number of classes held online, whilst still taking advantage of their ability to reduce the demand for physical space. More specifically, the problem is to investigate how universities can maximise the number of courses that they offer, whilst simultaneously limiting the number of online classes that are used to achieve this. This timetabling problem differs to the classic timetabling problem in that the input is a list of classes, but not all classes need to be assigned. Solutions to this problem identify how many courses could be offered, which is useful information for universities planning semesters. The preliminary model presented in this paper illustrates one way of solving this particular problem.

2 Model Formulation

2.1 Terminology and notation

Timeslots are lengths of time that have a start and end. In this problem we assume these are five minutes long. Timesets are defined as a subset of the set of all timeslots. These are used to better model complicated arrangements. For example, a timeset could describe a two hour class meeting every other week. Table 1 provides the notation for the sets used within the formulation of the model.

Table 1. Key notation. The first six sets are primarily used to describe elements of the problem, the last seven sets are primarily used in the construction of constraints.

S	Set of timeslots
T	Set of timesets. Each $t \in T$ is a subset of S
R	Set of rooms
C	Set of classes
K	Set of courses
L_k	Set of sections for course k. Each $l \in L_k$ is a subset of C
R_r^u	Set of timeslots when room $r \in R$ is unavailable
R_c	Set of rooms suitable for class $c \in C$
T_c	Set of timesets suitable for class $c \in C$
R_G	Let $G \subseteq C$. $R_G := \cap_{c \in G} R_c$
C_G	Let $G \subseteq C$. $C_G := \{(c_1, c_2) \in G \times G : c_1 \neq c_2\}$
R_r^c	Let $r \in R$. $R_r^c := \{c \in C : r \in R_c\}$
O_s	Let $s \in S$. $O_s := \{a \in T : s \in a\}$

We define the matrix A where the entry A_{r_1,r_2} is equal to the number of timeslots it takes to travel from room r_1 to room r_2. In particular, for d a non-negative integer, $A_{r_1,r_2} = d$ represents a travel time of $5d$ minutes.

The set C contains all classes regardless of if they can be held online, in-person or both. The online space is modelled as a room that is always available and can host multiple classes at the same time. Let r^* represent this online space. A class, $c \in C$, can be held online if and only if $r^* \in R_c$. For r^* assume that $A_{r^*,r^*} = 0$ and $A_{r^*,r} = d^*$ for all $r \in R \setminus r^*$ where d^* is a fixed number of timeslots. For this paper, we assume any class can happen online and that $d^* = 0$.

2.2 Main variables

The main variables used in this problem are binary variables indicating if a class is assigned to a particular room and timeset. They are defined as follows:

$$x_{c,r,t} = \begin{cases} 1 & \text{Class } c \in C \text{ is held in room } r \in R \text{ in timeset } t \in T, \\ 0 & \text{Otherwise.} \end{cases}$$

The second set of variables are auxiliary binary variables that indicate if a class uses a room, or if a class uses a timeset. They are defined as follows:

$$y_{c,r} = \begin{cases} 1 & \text{Class } c \in C \text{ is held in room } r \in R, \\ 0 & \text{Otherwise.} \end{cases}$$

$$y_{c,t} = \begin{cases} 1 & \text{Class } c \in C \text{ is held in timeset } t \in T, \\ 0 & \text{Otherwise.} \end{cases}$$

These variables are related to each other by linking constraints:

$$y_{c,r} = \sum_{t \in T} x_{c,r,t}, \quad \forall r \in R, c \in C,$$

$$y_{c,t} = \sum_{r \in R} x_{c,r,t}, \quad \forall t \in T, c \in C.$$

Helper arrays Defined in Table 2, helper arrays are fully determined by a given problem instance. They are used to indicate if a group of resources satisfy a particular condition, thus indicating what constraints to include in the model corresponding to that instance.

Table 2. Definition of each helper array. t, t_1 and t_2 are in the set T. r, r_1 and r_2 are in the set R

Array	Description
D_0	A matrix where $D_0[r,t]$ is equal to one if room r is unavailable at some point during timeset t, zero otherwise
D_1	An array where $D_1[r_1, r_2, t_1, t_2]$ is equal to one if there is not enough time between t_1 and t_2 to travel between r_1 and r_2, zero otherwise
D_2	A matrix where $D_2[t_1, t_2]$ is equal to zero if the first meeting in t_1 concludes before the start of the first meeting in t_2, one otherwise
D_3	A matrix where $D_3[t_1, t_2]$ is zero if the meetings in t_1 and t_2 do not occur on overlapping weeks and days, if any meeting in t_1 and t_2 occurs on the same day and week then $D_3[t_1, t_2]$ is equal to the number of timeslots between the start of the earliest meeting and end of the latest
D_4	A matrix where $D_4[t_1, t_2]$ is equal to zero if t_1 and t_2 start at the same time of day, one otherwise
D_5	A matrix where $D_5[t_1, t_2]$ is equal to zero if t_1 completely overlaps t_2 in the times of day they meet or vice versa, one otherwise
D_6	A matrix where $D_6[t_1, t_2]$ is equal to zero if t_1 meets on a subset of days that t_2 does or vice versa, one otherwise
D_7	A matrix where $D_7[t_1, t_2]$ is equal to zero if t_1 and t_2 do not meet on any of the same days, one otherwise
D_8	A matrix where $D_8[t_1, t_2]$ is equal to one if t_1 overlaps t_2, zero otherwise.

2.3 Constraints

There are various constraints that need to be included within any university timetabling model. Some are more specialised so that the timetable adheres to university policy or allows students and staff to travel comfortably between classes. In this section some constraints included within our model are described.

Classes can only be assigned at most a single room and a single timeset

$$\sum_{t \in T} \sum_{r \in R} x_{c,r,t} \leq 1, \quad \forall c \in C.$$

Classes can only be assigned compatible rooms and times To ensure only compatible rooms and times are used, for each $c \in C$ add the following constraints:

$$\sum_{r \in R} x_{c,r,t} = 0, \quad \forall t \in T \setminus T_c.$$

$$\sum_{t \in T} x_{c,r,t} = 0, \quad \forall r \in R \setminus R_c.$$

Classes should not happen in a room when that room is not available

$$\sum_{t \in T} \sum_{r \in R} D_0[r,t] x_{c,r,t} = 0, \quad \forall c \in C.$$

In-person classes should not use the same room at the same time

$$\sum_{c \in R_r^c} \sum_{t \in O_s} x_{c,r,t} \leq 1, \quad \forall r \in R \setminus r^*, s \in S.$$

Group of classes should occur in the same room Let G be a set of classes that must occur in the same room. For each $r \in R_G$ define a binary variable s_r^G that takes the value one if every class in G uses room r and zero otherwise. Add the following constraints:

$$\sum_{c \in G} y_{c,r} = |G| s_r^G, \quad \forall r \in R_G,$$

$$\sum_{r \in R_G} s_r^G \leq 1.$$

Attending a group of classes Staff and students have collections of classes they must attend. Denote a collection of classes as G. For each $(c_1, c_2) \in C_G$ add the following constraints:

$$D_1[r_1, r_2, t_1, t_2](x_{c_1,r_1,t_1} + x_{c_2,r_2,t_2}) \leq 1, \quad \forall t_1 \in T_{c_1}, t_2 \in T_{c_2}, r_1 \in R_{c_1}, r_2 \in R_{c_2}.$$

Group of classes should occur in a certain order Let G be a sequence of classes that should occur in order, meaning that the first meeting of a class should completely finish before the start of the next class. Suppose $G = (c_1, c_2, \ldots, c_k)$, then for each pair (c_i, c_{i+1}) where $i \in \{1, \ldots, k-1\}$ add the following constraints:

$$D_2[t_1, t_2](y_{c_i,t_1} + y_{c_{i+1},t_2}) \leq 1, \quad \forall t_1 \in T_{c_i}, t_2 \in T_{c_{i+1}}.$$

Group of classes should be grouped within a period of time Let G be a set of classes that should all happen within H timeslots if they happen on the same day and week. For each $(c_1, c_2) \in C_G$ add the following constraints:

$$I(D_3[t_1, t_2] > H)(y_{c_1,t_1} + y_{c_2,t_2}) \leq 1, \quad \forall t_1 \in T_{c_1}, t_2 \in T_{c_2},$$

where I is an indicator function that takes the value one if $D_3[t_1, t_2] > H$ holds and zero otherwise.

Timing constraints All timing specific constraints have the same form. Suppose G is the set of classes the constraint applies to. For each $(c_1, c_2) \in C_G$ add the following constraints:

$$D[t_1, t_2](y_{c_1,t_1} + y_{c_2,t_2}) \leq 1, \quad \forall t_1 \in T_{c_1}, t_2 \in T_{c_2},$$

where D is the appropriate helper array for that constraint. Table 3 describes a constraint on a group of classes and the associated helper array.

Table 3. Constraints and associated helper array

Constraint	Associated D
Classes should start at the same time	D_4
Classes should occur during the same time of day	D_5
Classes should occur on the same days of the week	D_6
Classes should occur on the different days of the week	D_7
Classes should not overlap in time	D_8

2.4 Objectives

There are many objectives in the timetabling literature. In this section some of the objectives that could be used within this model are presented.

Maximise number of courses offered Courses can be split into sections that teach identical content. Students attend exactly one of these sections if they are taking that course. Sections are a collection of classes and it is possible for two sections of the same course to share classes.

$$g_k = \begin{cases} 1 & \text{Course } k \in K \text{ is offered,} \\ 0 & \text{Otherwise.} \end{cases}$$

$$h_{k,l} = \begin{cases} 1 & \text{Section } l \in L_k \text{ of course } k \in K \text{ is offered,} \\ 0 & \text{Otherwise.} \end{cases}$$

For a given course, $k \in K$, a section $l \in L_k$ can be offered only if all of the classes in that section are offered. This is modelled by the following constraints:

$$h_{k,l} \leq \frac{1}{|l|} \sum_{t \in T} \sum_{r \in R} x_{c,r,t}, \quad \forall c \in l,$$

where $|l|$ is the number of classes in a section. A course is only offered if there is at least one section being offered. This is modelled by the following constraints:

$$g_k \leq \sum_{l \in L_k} h_{k,l}, \quad \forall k \in K.$$

To maximise the number of courses offered, we maximise the following objective function:

$$z_1 = \sum_{k \in K} g_k.$$

Maximise number of classes held Whilst courses are more important, offering as many classes as possible provides flexibility within these courses. To maximise classes offered we maximise the following objective function:

$$z_2 = \sum_{c \in C} \sum_{t \in T_c} \sum_{r \in R_c} x_{c,r,t}.$$

Minimise number of classes held online Ideally, there would be no need for online classes but it cannot be ignored that they can help increase z_1 because they are not subject to physical space limitations, a common limit to teaching capacity. To minimise number of classes held online we minimise the following objective function:

$$z_3 = \sum_{c \in C} \sum_{t \in T_c} x_{c,r^*,t}.$$

Minimise cost of assignment Let non-negative real numbers $P_{c,r}$ and $P_{c,t}$ be penalties for assigning class $c \in C$ to room $r \in R_c$ and timeset $t \in T_c$ respectively. The exact value of these penalties are subjective in practice. They could represent:

- Actual monetary cost of using the resource.
- Approximation of preference (low penalty indicating higher preference).
- Arbitrarily large penalties to deter solution from using a resource.

Cost of assignment to minimise is given as the sum of penalties:

$$z_4 = \sum_{c \in C} \left(\sum_{r \in R_c} P_{c,r} y_{c,r} + \sum_{t \in T_c} P_{c,t} y_{c,t} \right).$$

Trading-off multiple objectives Optimising z_3 and z_4 on their own will typically lead to a solution that assigns no classes at all. Therefore, it is important that z_1 or z_2 is optimised alongside these other objectives. One way this can be done is by maximising the following objective:

$$z = w_1 z_1 + w_2 z_2 + w_3 z_3 + w_4 z_4,$$

where w_1 is a positive weight, w_2 is a non-negative weight, and w_3 and w_4 are non-positive weights.

3 Results

To verify that this model produces feasible timetables, three instances from the 2019 International Timetabling Competition (ITC-2019) were used [4]. Described in Table 4 is the number of classes, timesets and rooms for each instance.

The problem defined in the ITC-2019 involves creating a complete timetable and allocating students to classes based on their course requests. Our model does not consider student allocation nor requires a complete timetable to be constructed, however, in this experiment we set w_1 and w_2 to one to encourage a complete timetable. We also set w_3 and w_4 to zero.

For our experiments we used an internal computing node running CentOS Linux with an Intel Xeon E5-2699 v3 CPU running at 2.30GHz and 528GB of RAM. The model was implemented in Python 3.5 and solutions were found using the commercial solver Gurobi 9.0, which successfully produced valid timetables. Table 4 provides information about each solution.

Table 4. The problem instances and quantities of key features. In the "Type" column, "T" indicates "test" instances and "C" indicates "competition" instances. The z_3 recorded is number of classes that does not require a room and are treated as "online-only" classes. The z_4 recorded here is the same value reported by the ITC-2019 validation tool

Instance	Type	$\lvert K\rvert$	$\lvert C\rvert$	$\lvert T\rvert$	$\lvert R\rvert$	z_1	z_2	z_3	z_4	Time (s)
lums-sum17	T	19	20	93	62	19	20	0	73	0.003
bet-sum18	T	48	127	50	46	48	127	6	3502	0.011
tg-fal17	C	36	711	1645	23	36	711	15	9610	58757.559

As can be seen from Table 4 in all three instances $|K| = z_1$ and $|C| = z_2$ meaning that our solutions are optimal for those objectives.

Acknowledgements We gratefully acknowledge the support of the EPSRC funded EP/S022252/1 STOR-i Centre for Doctoral Training.

References

1. Al-Yakoob, S.M., Sherali, H.D.: A mixed-integer programming approach to a class timetabling problem: A case study with gender policies and traffic considerations. European Journal of Operational Research **180**(3), 1028–1044 (2007). https://doi.org/https://doi.org/10.1016/j.ejor.2006.04.035

2. Barnhart, C., Bertsimas, D., Delarue, A., Yan, J.: Course Scheduling Under Sudden Scarcity: Applications to Pandemic Planning. Manufacturing & Service Operations Management **24**(2), 727–745 (2022). https://doi.org/10.1287/msom.2021.0996, https://doi.org/10.1287/msom.2021.0996
3. Lederman, D.: What's the future of the physical college campus? Inside Higher Ed (2021), https://www.insidehighered.com/news/2021/07/16/whats-future-physical-college-campus
4. Müller, T., Rudová, H., Müllerová, Z.: University course timetabling and International Timetabling Competition 2019. Proceedings of the 12th International Conference on the Practice and Theory of Automated Timetabling (PATAT 2018) pp. 5–31 (2018)
5. Vermuyten, H., Lemmens, S., Marques, I., Beliën, J.: Developing compact course timetables with optimized student flows. European Journal of Operational Research **251**(2), 651–661 (2016). https://doi.org/https://doi.org/10.1016/j.ejor.2015.11.028