# Matheuristic for Approximating a Frontier for a Many-objective University Timetabling Problem

Matthew Davison[1], Ahmed Kheiri[2], and Konstantinos G. Zografos[3]

[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@lancaster.ac.uk
[3] Department of Management Science, Centre for Transport and Logistics (CENTRAL), Lancaster University, Lancaster, UK
k.zografos@lancaster.ac.uk

**Keywords:** University Timetabling, Hybrid Teaching, Mixed Integer Programming, Matheuristic

## 1 Introduction

The University Course Timetabling Problem (UCTTP) involves assigning *events* (such as lectures and seminars) to *times* and *locations* along with assigning *staff* and *students* to these events. One approach to formulating this mathematically is to model the problem as a Mixed Integer Linear Program (MILP) [1]. Given the resulting complexity of the problem for real-life instances, heuristic approaches have been proposed to solve the problem [2].

One matheuristic used is known as fix-and-optimise [4] where neighbourhoods are allowed to improve while the rest of the solution is unchanged. Typically, this has been used as part of a single-objective approach. However, the UCTTP can be modelled as a multi-objective problem. Trade-offs in conflicting objectives can be found using an $\epsilon$-constraint approach but this is hard when working with many objectives.

In a decision support context, it is desirable to quickly find these trade-offs by generating an approximation(s) of the Pareto frontier(s). Generating high-quality frontiers allows decision-makers to understand the trade-offs so the most desirable timetable for implementation can be selected.

We propose a method that leverages the gradual improvement of fix-and-optimise to search the objective space but still allows for elements of $\epsilon$-constraint approaches to be added and removed when needed. This results in a matheuristic approach suitable for any many-objective problem.

## 2 Problem description

The details of the specific UCTTP we are solving here are described in [3]. The key aspect of the UCTTP in this paper is incorporating hybrid teaching, where classes can happen in-person, online or in a hybrid mode. In this model, students can express a

preference for a certain mode of study. For this reason, we focus on the following three objectives:

$z_1$: Maximise total module attendance.
$z_2$: Minimise total number of deviations from mode preferences.
$z_3$: Minimise total number of student scheduling issues.

## 3 Method description

The method requires at least one feasible solution as input. We understand that this itself is not a trivial task for any UCTTP however many techniques have been proposed [2]. We will assume that the (approximate) nadir point is known. The flowchart in Figure 1 outlines the general structure of the method.

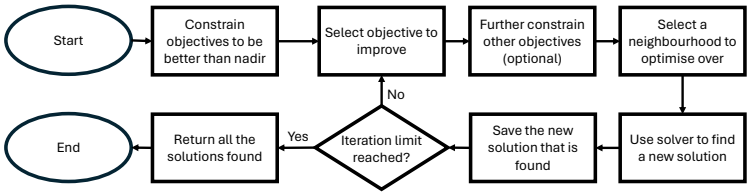


Fig. 1: Flowchart providing an overview of the method.

The pseudocode presented in Algorithm 1 describes a simple variant of this method. There are two comments on this pseudocode. These indicate the key places where more sophisticated selection methods could be used instead of random selection. The combined use of neighbourhoods and unconstrained objectives allows for small changes in objectives like an $\epsilon$-constraint approach.

## 4 Method demonstration

The instance for this demonstration is a modification of the instance *mary-fal18* from the fourth International Timetabling Competition [5]. The significant changes are that we only consider 400 students and reduce physical room capacity fourfold (for details see [3]). The initial solutions are four lexicographic solutions (see Table 1) and using the objective values for these solutions we can obtain the exact nadir point. The method used for the demonstration is described in Algorithm 1. We have $|S| = 4$ and use the parameters $I = 50$, $R = 5$ and $N = 99$. This produces a new pool of solutions $S^{new}$. Any solution in $S^{new}$ dominated by another solution in this set is removed. The results were found with a machine with Ubuntu 22.04.1 LTS using an Intel(R) Xeon(R) Gold 6248R CPU running at 3.00GHz and 16GB of RAM. The method was implemented in Python 3.10.12 using Gurobi 10.0.2.

This run of the method yields 68 non-dominated alternative solutions. It can be shown that some of the non-dominated solutions were found after visiting what would turn out

**Algorithm 1:** Pseudocode for simple variant of method

**Input:** Set of initial solutions $S$, # iterations $I$, # repeats $R$, neighbourhood size $N$.
**Output:** New set of solutions $S^{new}$.
$S^{new} \leftarrow \{\}$;
Constrain all objectives to be better than the nadir point;
**for** $r = 0$ *to* $R$ **do**
  **for** $s$ *in* $S$ **do**
    $s^{current} \leftarrow s$;
    **for** $i = 0$ *to* $I$ **do**
      Randomly select objective $z$ to optimise ; `// Selecting objective`
      Randomly select $N$% of students to fix; `// Selecting neighborhood`
      Fix selected students according to $s^{current}$;
      Optimise $z$ to find $s^{new}$;
      $S^{new} \leftarrow S^{new} \cup \{s^{new}\}$;
      $s^{current} \leftarrow s^{new}$;
    **end**
  **end**
**end**
**return** $S^{new}$;

Table 1: All lexicographic orderings of objectives and their corresponding objective values. Some orderings attained the same objective values.

| Ordering(s) | $z_1$ | $z_2$ | $z_3$ |
|---|---|---|---|
| ($z_1$,$z_2$,$z_3$), ($z_1$,$z_3$,$z_2$) | 1,599 | 102 | 53 |
| ($z_2$,$z_1$,$z_3$) | 1,506 | 0 | 26 |
| ($z_2$,$z_3$,$z_1$), ($z_3$,$z_2$,$z_1$) | 1,486 | 0 | 0 |
| ($z_3$,$z_2$,$z_1$) | 1,554 | 68 | 0 |

to be a dominated solution. Figure 2 shows that the alternative solutions stay close to the starting solutions and that some starting solutions produced more alternative solutions than others. Only certain sections of the efficient frontier have been approximated but this may be improved using more sophisticated objective and neighbourhood selection procedures.

## 5 Future work

There are several possible research directions to take this work in:

- Develop better objective and neighbourhood selection procedures so that more of the frontier can be approximated.
- Compare the generated frontiers with frontiers found using an exact method to assess the quality of the method's output.
- Validate the quality of this method by using a large test set with a wide range of instances.

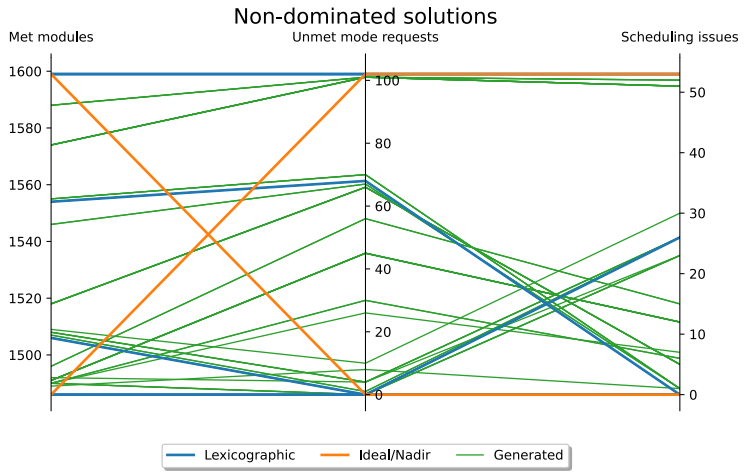


Fig. 2: Plot of non-dominated solutions and their corresponding objective values.

- Experiment with real-life instances rather than using instances that have been reduced in size.
- Apply the method to a variant of the UCTTP with more than three objectives.

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

## References

1. Aziz, N.L.A., Aizam, N.A.H.: A brief review on the features of university course timetabling problem. AIP Conference Proceedings **2016**(1) (2018). https://doi.org/10.1063/1.5055403, https://aip.scitation.org/doi/abs/10.1063/1.5055403
2. Babaei, H., Karimpour, J., Hadidi, A.: A survey of approaches for university course timetabling problem. Computers & Industrial Engineering **86**, 43–59 (2015). https://doi.org/https://doi.org/10.1016/j.cie.2014.11.010, https://www.sciencedirect.com/science/article/pii/S0360835214003714
3. Davison, M., Kheiri, A., Zografos, K.: Modelling and solving the university course timetabling problem with hybrid teaching considerations (Under Submission, 2024)
4. Mikkelsen, R.Ø., Holm, D.S.: A parallelized matheuristic for the International Timetabling Competition 2019. Journal of Scheduling **25**(4), 429–452 (Aug 2022). https://doi.org/10.1007/s10951-022-00728-8, https://doi.org/10.1007/s10951-022-00728-8
5. Müller, T., Rudová, H., Müllerová, Z.: Real-world university course timetabling at the international timetabling competition 2019. Journal of Scheduling (Apr 2024). https://doi.org/10.1007/s10951-023-00801-w, https://doi.org/10.1007/s10951-023-00801-w