Timetable Optimisation Final Report

Ben de Sousa, Todd House, Alexandros Zachakos, Nolan Willoughby, Conor McArthur

Department of Mathematics
University of Edinburgh

May 6, 2026

Table of Contents

1 Introduction

The University of Edinburgh Timetabling team has presented the problem of optimising the university timetable for future years. It is a challenge which the team undertakes every year and they are always looking for ways to improve on previously generated timetables. This report will tackling the time scenario set of questions that team had presented. This set focuses on the possibility of constraining the hours in which the university operates. In particular four main tasks, will be tackled:

- A reduction in core hours
- An enforcement of a lunch break for students with >4 hours of contact time in a day
- Minimising clashes
- A reduction in idle times for rooms

Success for this project will be measured by the extent at which these targets will be met.

1.1 Background

Development of the timetable is a major challenge that the university faces every year. The many programmes offered by the institution have a large variety of event types, contact hour requirements and room requirements. For example, certain courses within the Edinburgh College of Art may require the use of specialised production studios for certain parts of their teaching. Comparatively programmes within the School of Mathematics may not have this particular requirement however, they require much higher contact time for the students. Adapting the timetable for these specific requirements introduces large computational burdens so goal of this project is to identify a new approach which is both easier to implement and produces a more optimal final timetable.

The Timetabling team aims to avoid creating situations such as lectures being scheduled late into a day, clashing between core lectures for a course and many situations beyond these. Currently, the team primarily uses the Timetabler tool produced by EventMap (Optime Timetabler [1]) for the majority of the computational work for this. EventMap itself uses a proprietary algorithm, created by its founders (McCollum et al. [2]), based on the Great Deluge Algorithm (GDA). The GDA being a metaheuristic which employs a steadily tightening boundary condition to converge towards an acceptable solution. While this clearly functions well for its intended purpose, this report intends to investigate whether an alternative approach may present a better, more sustainable solution.

1.2 Literature Review

Historically, constraint programming was the standard approach to large-scale timetabling problems. This approach has specific advantages with regard to the structure of timetabling constraints, the objective of the problem, and the tendency of constraint programming solvers to efficiently prune the solution space of a problem. Additionally, Cambazard et al. demonstrate an extension of constraint programming allowing for quick adaptation of solutions to dynamic aspects of the university timetabling problem including linking different timetables with shared resources and lecturer requirements, especially in the case of adjunct lecturers. Their approach involves computing explanations, information that justifies the reduction of variable domains, as part of a three step process that adjusts solutions as constraints are either dynamically added or retracted by the person generating the timetable (Cambazard et al. [9]). While this approach demonstrates promising results in computational complexity and usability, constraint programming approaches do not provide guarantees of solution quality as with other optimization methods used in the domain of timetabling, notably mixed-integer programming.

The improvement of modern mixed-integer programming (IP) solvers have led to the further utilization of IP as a method for timetabling. An example of this is the approach used by Aygül et al., who introduced a stepped IP approach to generate an initial, feasible timetable then minimize the number of relevant scheduling conflicts under various enrolment scenarios by allowing course sections to be dynamically split, subject to location, time, instructor, and splitting logic constraints. In this case, the solution space was already quite constrained by the number of considerations and the structure of the university (undergraduate courses are taught on a quarter-based schedule while postgraduate courses are taught on a semester-based schedule while sharing resources, the university is a university of technology and has a relatively small enrolment, and programme structure tends to "common triples" of course choices by students). In spite of this, computational experiments on university data conducted on a computational cluster with 128 GB of RAM running Ubuntu 20.04.06 generated required runtimes of up to 8.6 hours to produce solutions of satisfactory optimality (Aygul et al. [10]). While this approach is therefore suitable for certain applications, its computational load is such that the introduction of more efficient methods is of interest, especially as this specific use case involves a problem with much greater scale.

Modern solvers such as Gurobi implement heuristics to more quickly find feasible solutions with potential for improving the objective value (Gurobi LLC. [11]). However, as in the example of Aygül et al., these heuristics often insufficiently reduce the computational load for certain use cases. The initial approach to the problem will be to utilize IP. Historically, timetabling optimisation problems have been solved through the use of Integer Programming as well as heuristics (Lamas et al. [3]). The IP typically performs the bulk of the computational optimization. The heuristics are used either to reduce the search space, to generate initial feasible solutions, or as stand-alone solvers when IP becomes computationally intractable at scale. Because of their nature general heuristics can lead to solutions that are somewhat arbitrary depending on the criteria used. However, they became more necessary as problems began to scale upwards over time. Matching this trend newer, more refined heuristics began to be developed and deployed to produce ever improving results. One such example is the class of Genetic Algorithms (GAs).

Recent approaches have used such GAs (Shah et al. [4]) to generate timetabling solutions within the education sector. In these cases there is high success in tackling the larger scale problems that IP struggles with however, by their nature GAs do not guarantee optimality in the same way as an IP model. This limitation motivates a hybrid approach adopted later in this project. Based on the work seen in other recent papers (Arias-Osorio and Mora-Esquivel [5]), which demonstrate the effectiveness of a hybrid approach. A GA metaheuristic will be used in conjunction with an IP model. This should allow for the problems of both scale and optimality to be addressed. A GA will be used to guide the search while the IP model will improve the quality of solutions within the feasible regions identified.

2 Mathematical Formulation

We consider any given programme that is 1-year or longer and each distinct event in a course. Courses are broken down into compulsory and optional categories. Rooms are distinguished by number of seats available. We seek to avoid overlapping compulsory courses in the same programme and year, overlap between optional and compulsory, if core teaching hours can be constrained to Monday-Friday 9-5pm, eliminating teaching on Friday with no other change to teaching hours, if all clashes can be avoided, and if the majority of students could have a one-hour lunch break during a 12-2pm window.

The full mathematical model formulation can be found in the Appendix.

2.1 Constraints

Extension Constraints

These a set of bespoke constraints that are being considered for implementation. If possible they will be included within the model however, these will also be the first omitted when hitting practical limitations. We would like to consider whether the timetable is likeable by both the students and the teaching staff, independent of its feasibility.

Our first consideration is that it might be more convenient to have consecutive events for the same course, for example 2-hour lectures. The generalization of it for n consecutive hours can be formulated as:

$$x_{i,t,d} \leq x_{i,t+1,d} + x_{i,t-1,d}$$

$$2x_{i,t,d} \leq x_{i,t-2,d} + x_{i,t-1,d} + x_{i,t+1,d} + x_{i,t+2,d}$$

$$\vdots$$

$$nx_{i,t,d} \leq \sum_{r=1}^{n} (x_{i,t-r,d} + x_{i,t+r,d})$$

$$\forall i \in I_k,\ k \in K,\ t \in \{2, \ldots, 8\},\ d \in D$$

It should be noted that the $\forall i \in I_k$ can be violated depending on the preferences of each programme.

What we also want to consider is that the daily teaching load is less than 6 hours so students are able to fully concentrate during the whole day, thus:

$$\sum_{t \in T} \sum_{i \in I_k} x_{i,t,d} \leq 6 \quad \forall k \in K,\ d \in D$$

3 Data Preparation

The Timetabling team provided a number of datasets to aid in tackling the various questions that they provided. Selection and processing of the correct datasets was critical to ensuring the validity of the results.

3.1 Dataset Selection

The initial dataset selected from the 2024-5 Event Module Room data. Any events from the Module Department listed as Royal (Dick) School of Veterinary Studies were used to trial and conduct diagnostics. This subset provided us a smaller amount of courses and locations to consider while building our model. The veterinary school is also the most isolated group of courses and locations, where they have very little or no overlap with other schools and locations. After validating the model on this smaller subset, it was possible to then apply it to more schools. Following validation of the model against the initial subset, we then tested it against all schools listed in the Event Module Room data.

3.2 Cleaning

The data provided is extensive and quite messy. This necessitated additional steps to clean the data for use. Much of the data is understandable from an administrator's standpoint, but not easy to comprehend for external actors. The primary issues were: "encoded" or abbreviated naming conventions, individual lectures for the same course at the same time and location were listed, and conflicting indicators regarding room requirements and location. We separated out the year group from the module name, collapsed courses down to one entry per time slot and location, and amended indicator conflicts.

4 Implementation of the Model

The mathematical formulation was implemented in Gurobi with Python. This process follows the previously mentioned pipeline that increased the complexity and scope of the model at each stage.

4.1 Testing on the Veterinary School

Initially the feasibility of the model was tested on the smaller Veterinary School Dataset. This generated results quickly and with a high degree of optimality without much additional tuning being required. In fact it reached a MIP Gap of 0.00% indicating that the timetable produced contained the minimal number of clashes possible. To visualise the result of running the model on this dataset we can examine the heatmap timetable below,

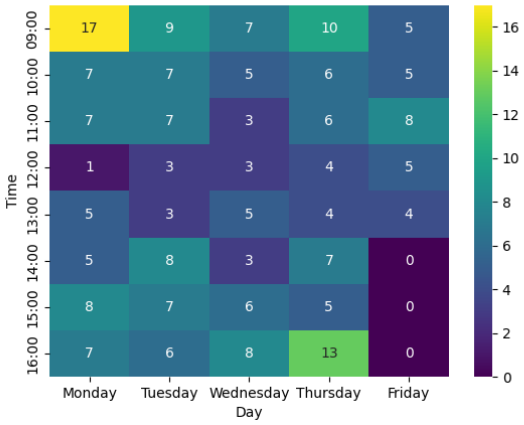


Figure 1: Heatmap of the Veterinary School Timetable

The concentrations of the events is relatively evenly spread - with the notable areas of high concentration being Monday and Thursday morning as well as Thursday afternoon. These are the times selected for optional, not Whole Class, courses as the clashes are allowed between these events. It can also be seen that the lowest concentrations of events are around the imposed 12:00-14:00 lunch break period. This follows from the fact that our implementation of a lunch break merely requires that each programme has one hour off each day in that period. While it is not possible to completely remove events from this period, the low concentration suggests that is likely that most students will have no event scheduled at some point within it. Also of note is that no events occur on any day between 17:00 and 18:00, nor on Friday afternoons after 14:00.

The success of this model justifies subsequent testing on the full dataset for the university courses.

4.2 Expanding to the Full Dataset

With proof of concept from the testing on the Veterinary School the next stage was to move to running the model with the full dataset to generate a timetable for the whole university.

4.2.1 Dataset Insight

On the initial attempts to run the model on the full dataset, it was discovered that there were further issues with the dataset that need to be addressed for a feasible solution to be found. There were four major issues which needed to be addressed:

- School Specific Timetabling issues

  - At least one Medical School programme and potentially more programs across more schools required >40 hours of lecture time per week. This is beyond the maximum hard limit that was being imposed when considering the constraints related to the Timetabling Office's questions.
  - The Futures Institute seems to require members of its programs to take a set of required courses with each course taking place over a unique set of weeks. Given our consideration of each course's demand being equal its highest recorded demand over any week in which it was scheduled, this led to programs again requiring >40 hours of lecture time per week.
  - At least one programme in the School of Mathematics and potentially more programs across more schools offered too many optionals to schedule all optional and required courses for their respective demands without conflict.

- Programmes using the same slot for a new event each week
- Tutorials being assigned far higher demand than what was required
- Several events exist in the data with a corresponding event size of 0.

These latter three issues and some school-specific issues were generally addressed by manually adjusting data values on a case-by-case basis as they appeared as infeasible constraint systems when implemented in Gurobi. School-specific issues tended to require further consideration, as the nature of those issues tended to be more structural than data values with unclear interpretation.

Specifically, relaxations of base model constraints needed to be made. Because of the lack of clarity in their programme structures and requirements, all Futures Institute courses had demand set to 0, effectively removing the Futures Institute from consideration. Due to a lack of mapping programme IDs to programme names, actually removing these courses and programs from the model entirely was not possible, and Gurobi's presolving methods were relied upon instead. Constraints limiting the teaching week to 40 hours had to be relaxed to allow programs with data indicating their lecture demand exceeds 40 hours to be timetabled.

This naturally leads to a model that is largely relaxed from the base case (detailed at 11.2 in the Appendix) however, it should still serve well as a representative of the potential solutions that could be generated.

4.2.2 Results

Unfortunately, due to the much higher complexity and computational demand of generating the timetable for the whole university, the limits on the resources available for this project lead to a timetable with a number of penalties and notes that need to be attached to it. Below is a heat map of the timetable for the full university, obtained by running the relaxed model in Gurobi for ten minutes.

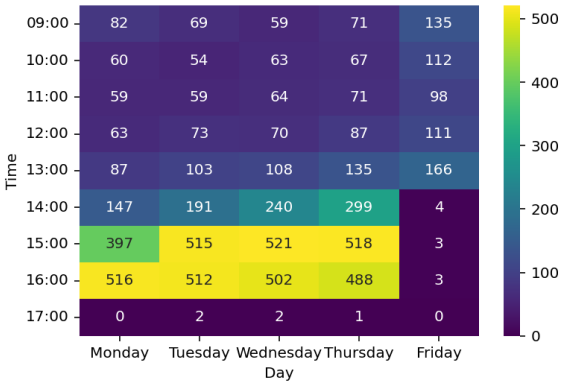


Figure 2: Heat map of the Full University Timetable

Events are highly concentrated in the afternoon, and indeed anecdotal evidence suggests there exist many multi-way clashes occurring in the afternoon. Overall, two programs violate the rule requiring teaching be limited to 09:00-17:00 (one twice and the other thrice), five programs violate the rule that Friday afternoons must be open (one once and the rest thrice), and 1095 programs violate the constraint limiting possible teaching hours to six per day (violations range from one hour each week to 185), totalling to 33268 hours scheduled in violation of the rules.

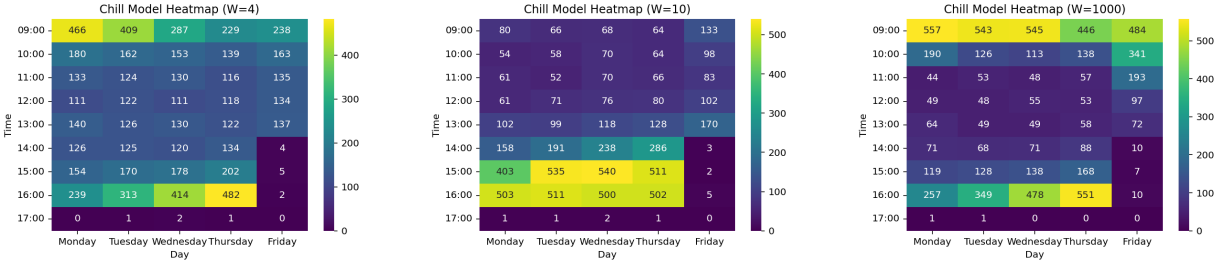
With an objective value of $\approx$ 127600 and a weighting where each clash contributes a factor of ten to the objective, this implies the proposed timetable has $\approx$9400 clashes. It should be noted that these results correspond to a MIP Gap of nearly 60% (based on later implementations this solution has an actual optimality gap of $\approx$50%), and observationally it is clear that better solutions exist that involve more events scheduled in the morning. Obtaining these solutions requires the implementation of improvements to solution method.

It can be noted that this conclusion was only settled on following a 47 hour run of this baseline model to attempt to achieve optimality. In the end the MIP Gap was 5.19% indicating that the MIP was converging towards a solution. However, it was clear that a more efficient approach would be required.

4.2.3 Sensitivity Analysis

Tuning the penalty parameter W yields different results. As $W \to 0$, the relaxation is dis-incentivized to avoid clashes, so any timetable that does not violate the given rules is optimal regardless of feasibility for students. So while reducing W leads to quicker convergence to optimality, the quality of timetable produced likely degrades as well.

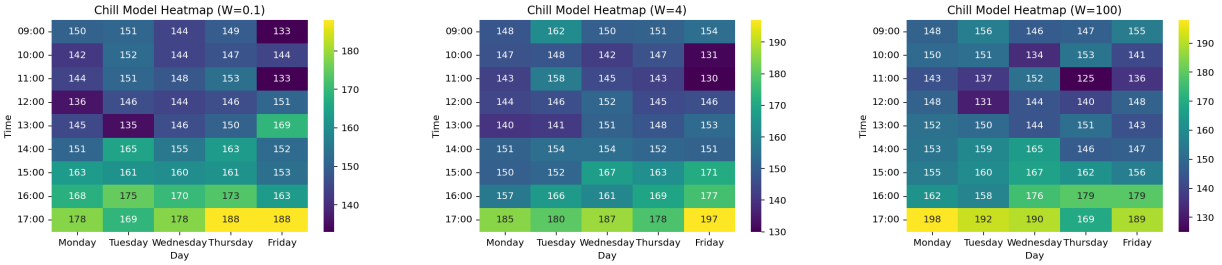
When considering all the extra rules that we want to enforce in the timetable, and running the model 3 times for 10 minutes and different values of W, we get the following results:



Unfortunately, this outcome comes with the respective limitations in terms of optimality, as in any case the MIP gap is relatively high. However, it was noticed that for smaller values of W, the MIP gap

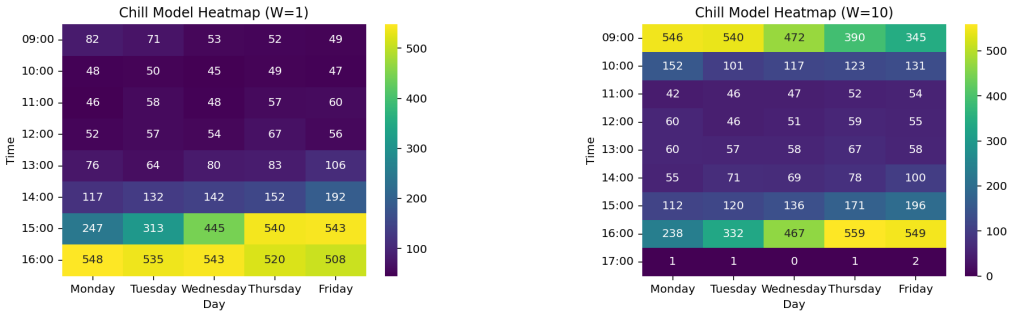
dropped as well, providing a range from approximately 3% ($W = 4$) to 80% ($W = 1000$). Despite the unsatisfactory results, at these levels of optimality, the solver accounted for all of the relaxed rules that were enforced. The most apparent proof of that is the limited number of events being scheduled after 5pm, and on Fridays afternoon, with a slight increase being recorded the less we penalize the violation of the respective rules, i.e. the increase of W.

In case we want to account only for providing a lunch break between 12pm and 2 pm, we get the following results:



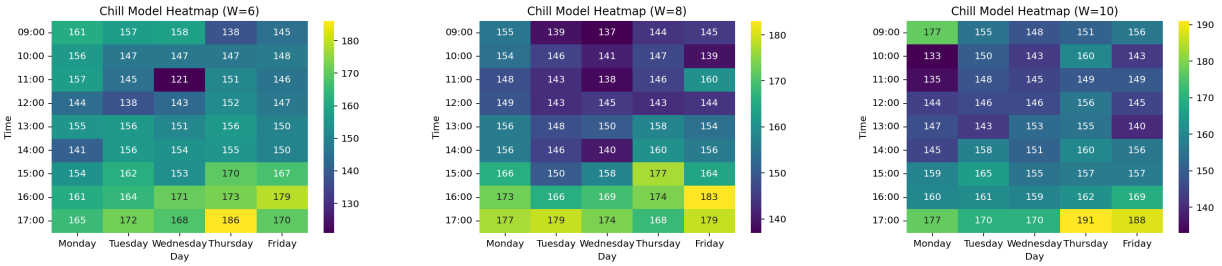
The difference for each timetable for different values of W is not significant, again based on the optimality limitations of the MIP gap, and a similar cumulation of events across all scenarios is being provided.

What can also be done is the inclusion of weights on each type or rule we are enforcing. In this instance, penalties for events being on Friday afternoon will not be considered, while we weight the penalties for no lunch break, events after 5pm and more than 6 hour-long days with 10, 0.001 and 1 respectively.



In that case, it can be seen that the time period for lunch breaks is one of the least condensed periods in the timetables, which makes sense as the respective rule was one of the main considerations. Also, it is apparent that for smaller values of W, where we penalize rule breaking the most, all events after 5pm have been rescheduled for another time, despite that rule having the smallest weight of all. This fact indicates that when we penalize for less rules (no penalty for Friday afternoon events this time), there is a better chance of achieving the total enforcement of the remaining ones, even when they are considered in a negligible extend.

Finally, we shall consider the outcome of the timetable in case we penalize none of the rules given, aiming entirely in the construction of a uniform timetable:



Now, solving for different values of W, does not any new information regarding the impact of the enforced rules, or at least in a theoretical scope. Nonetheless, because of the MIP gap and the solution process each time, the solver provides us with different timetables being found in the span of the 10-minute run time period. In any case, the lack of extra constraints has resulted in more uniform timetables, further ensuring the reliability and the potential of the base algorithm, even in cases of non-optimality.

5 Extending the Model

5.1 Extension Algorithm

In order to extend the model and help the solver provide better results faster, we are going to use a Genetic Algorithm Heuristic. The general structure of the GA Heuristic is the following:

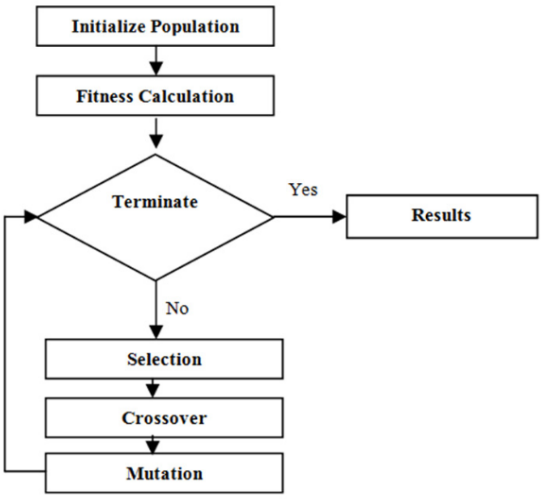


Figure 7: GA flowchart[1]

This algorithm shall run two times. The first time is in order to find a set of feasible schedules that satisfy all hard constraints (Alg1), and the second one is in order to minimize the total number of clashes derived from the soft constraints (Alg2). The steps of the algorithm that we used are the following:

1. (Alg1): Randomly create a population of n potential timetables.
   (Alg2): Consider the population derived from the implementation of (Alg1).

2. Assess all the timetables of the population, and list them according to the total number of violations (Alg1) / clashes (Alg2), from the one with the least, to the one with the most

3. Choose two of the candidate schedules to crossover. Those with less violations/clashes have a higher chance of being chosen as the "parents". The probability for timetable i to be selected

as a parent is:

$$\frac{V_{max} - V_i}{V_{max}}$$

where V_{max} is the maximum number of violations (Alg1) / clashes (Alg2) in the population, while V_i is the number of violations (Alg1) / clashes (Alg2) of timetable i.

4. Crossover aspects from both "parents". In particular, if for course i we have $x_{i,t1,d1} = 1$ from one parent and $x_{i,t2,d2} = 1$, then the possible outcomes for the new timetable are:

$$\begin{cases} x_{i,t1,d1} = 1, & \text{with probability 1/4} \\ x_{i,t2,d1} = 1 & \text{with probability 1/4} \\ x_{i,t1,d2} = 1 & \text{with probability 1/4} \\ x_{i,t2,d2} = 1 & \text{with probability 1/4} \end{cases}$$

5. Given a probability p, decide if a mutation will occur in the new timetable or not. In that case, there are two types of mutation that we can use, and they can happen equally likely:
   (a) Choose randomly a course i with $x_{i,t,d} = 1$, a time t^*, and a day d^*. Then set $x_{i,t,d} = 0$ and $x_{i,t^*,d^*} = 1$, in order to randomly assign the course somewhere else in the timetable.
   (b) Choose randomly two courses i and j with $x_{i,t1,d1} = 1$ and $x_{j,t2,d2} = 1$. Then set $x_{i,t1,d1} = x_{j,t2,d2} = 0$ and $x_{i,t2,d2} = x_{j,t1,d1} = 1$ in order to switch the places of the two courses on the timetable.

6. Check if the new timetable satisfies all hard constraints. If not, we use a local search heuristic to take a feasible solution. We first identify the violations and then we randomly choose one of the courses involved and move it somewhere else like demonstrated earlier. This procedure is repeated until no hard constraint violations are left in the timetable.

7. Assess the new timetable based on its violations (Alg1) / clashes (Alg2). If it is better than the worst timetable in the population, then replace the bad timetable with the new one. Otherwise change nothing and proceed. If Λ iterations pass and a feasible solution for the new timetable is not found then reject it and go back to step 3.

8. (Alg1): Repeat until all the timetables in the population are feasible with zero violations.

   (Alg2): Repeat until the same timetable is considered the best one of the population for K consecutive iterations, indicating that its difficult for the algorithm to find an even better solution than the best one so far.

The process for implementing the model was relatively straight forward. It required constructing the logic of the GA and then inserting the baseline model into the requisite steps. That being to first run the baseline MIP to ensure a quality level in the final solution, then this would be run through the GA before finally running the MIP one final time.

5.2 Extended Model Results

On initial runs of this model, with a 10 minute time limit on the baseline, there was a 4% improvement in MIP gap validating the theory that the GA would help to improve the results. The next step was to attempt to reach optimality with a longer run of the model.
In practical terms for this experiment, due to computational and time constraints, the final solution was obtained under limiting conditions. These being; limiting the initial MIP run to a 19 hour run time, constraining the GA to a population size = 20 and generations = 50, and finally setting the time limit of the final run of the MIP to 10 hours. Despite these limitations the results obtained from this extended model showed promise.
Below is the final heat map timetable obtained from this extended model. It should also be noted

that this was the only long run of this model, subsequent pieces of analysis were carried out with significantly shorter time limits to ensure that some data could be collected for this report.

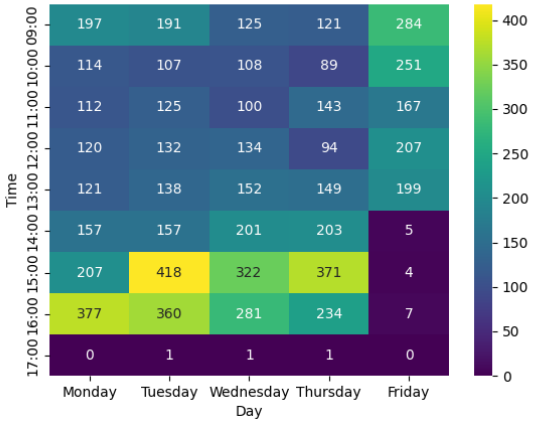


Figure 8: Heat map of the Extended Model Timetable

Visually it is already clear that the quality of the timetable solution achieved has improved. Events are more evenly spread and there are half the number of violations of the 9-5pm constraint. The peak number of events carried out in a given slot for this timetable is 418 whilst it is 521 for the baseline model. There are an increased number of events violating the Friday afternoon restrictions however, this would seem to be a worthwhile trade-off for overall improvement to the timetable.
Examining the MIP gap and time frames, this extended model reach a final gap of 46.81% in 29 hours which compared to the initial baseline model run is actually worse. The baseline model that fed into the GA actually achieved a gap of 43.8% in 19 hours. This would indicate that the GA implementation led to a decrease in performance. However, there are some interesting things to note in the heat map of the baseline model below.

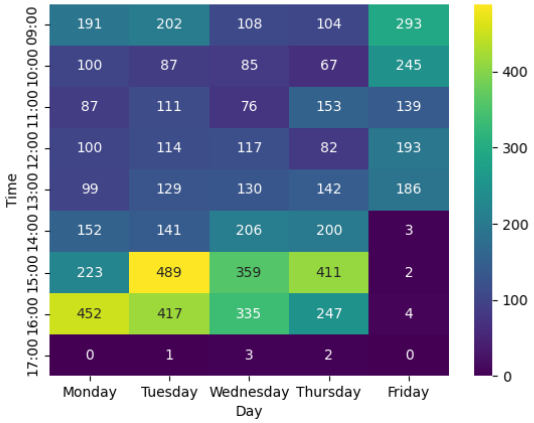


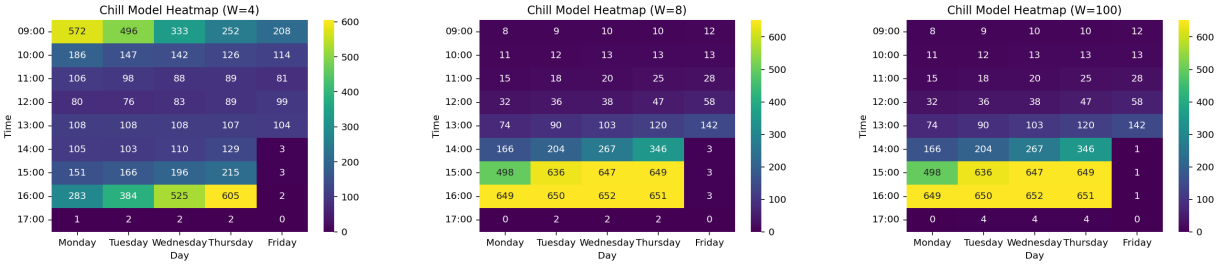
Figure 9: Heat map of the Long Run Base Model Timetable

The baseline model shows double the violations of the extended model for the 9-5pm constraint whilst also displaying greater clustering of individual events in single time slots, peaking at 489. So, this would indicate that although the extended model resulted in a worse MIP gap, the actual quality of the timetable produced improved. However, it is also apparent that despite the GA being implemented to improve the efficiency of the model, it cannot be concluded that it does so based on these results. With greater compute time there may have been greater improvements displayed. This implementation does still show promise regardless and would be worth further investigation.

5.2.1 Sensitivity Analysis

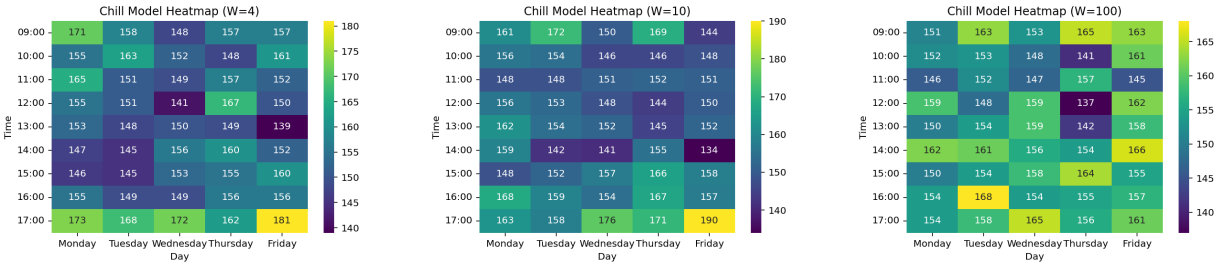
In comparison with the original model, solving the extended version is even more time-restrictive, as the number of computations increases due to the inclusion of the Genetic Algorithm. This further limits sensitivity analysis as when running for 20 minute for each value of W, the solver often returns almost identical results.

In particular, when we account for all the given rules, and for the same weights, we get the following results:



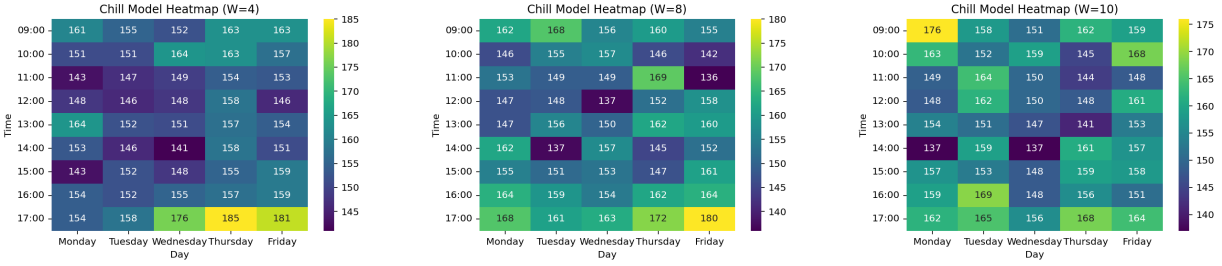
As it can be seen from the graph for $W = 4$, the result with the genetic algorithm has been improved, and now there exists a clear lowest concentration of events in the lunch break period, while maintaining the low number of events pattern after 5pm and on Friday afternoon. Moreover, if we increase the value of W from 8 and above, we notice that the solution is almost identical, indicating the failure of the algorithm to find any different solutions in the time given.

Now, we want to see what happens with the extended model when only the deprivation of a lunch break is penalized.



As expected, the algorithm accounts for the lunch break in all cases, especially when W is low. The lunch break hours are once again among the ones with the lowest number of clashes, while the increase of W, makes the events to look more and more uniformly allocated across all possible time-slots.

Finally, we want to check the extended model as its own without any additional rules for distinct values for W as well:



The results are once again a lot clearer and more valid than the ones where rules where applied. Again, running the model for different values of W in this instance, doesn't really provide us with

any additional information about the consideration of different features, but it enables us to derive a variety of new timetables and compare them with their predecessors in the original model, visualizing the better performance of the algorithm in the extended form.

6 Final Model Evaluation and Recommendations

Having implemented the model, the following summarizes considerations for model implementation and recommendations to improve model results.

6.1 Practical Implementation

Given the complex nature of the schedule, the extensive courses to consider, and the difficulty of strictly constraining the problem, it would be challenging to practically implement this in the University Timetabling Department. Use of this tool would likely require a specialised individual capable of running this and tuning it multiple times at the beginning of the semester. This would need to be done in parallel with the current methods of timetable development employed by the university. It would be impractical to iteratively use this for minor adjustments throughout a given semester.

Another significant hurdle would be the operational debt required to implement the tool. It would likely take at least 3 semesters of implementation to potentially see some reduction in time demands to build the timetable for the university. Given the strained resources in the department, it appears unlikely that this would be a worthwhile investment to dedicate labour hours towards.

6.2 Recommendations

This report recommends further refinement of the method to achieve better results and a more practical use case. Multiple options exist that could provide a better suited structure for OR solutions. Removing the need for every lecture having a unique event if it is for the same course, at the same time, and in the same location would greatly reduce the data processing demands and prevent errors. Certain events that are non-standard teaching formats, such as week-long drop-in practicals, should be removed and manually processed. We referenced Sherali and Driscoll [7] to try to understand what aspects of the problem space provided effective results.
If enough organisational backing could be achieved, there exist major methods of overhauling teaching to provide better results.

1. Tiered Deconfliction Meetings: Given that each school already develops an internal timetable without room assignments, it would be relatively easy to consider implementation of tiered deconfliction meetings. This process would follow the current structure, except schools would also allocate rooms to each given lecture and workshop. After compiling and deconflicting at the school level, representatives from each school could participate in a university-wide deconfliction meeting where conflicts are resolved on a case by case basis. This would be done by individuals with more refined knowledge of the needs of the courses, the lecturers, and the number of cross-school students that typically attend.
   It would make sense to have highly linked schools, e.g. the School of Literatures, Languages, and Cultures with the School of History, Classics, and Archaeology, do joint deconfliction before sending tables up to the timetabling team. The schools with the most crossover participation can be seen in Figure 13. The specific courses with the most crossover enrolment are exclusively year 1 and year 2 courses. Some smaller courses see up to a 7:1 ratio of outside enrolment to internal. Even larger courses, for example Computer Science and Mathematics (BSc Hons) (Year 2) had 171 students with a ratio of 4.5:1. These suggest that perhaps too much choice exists in these early years.

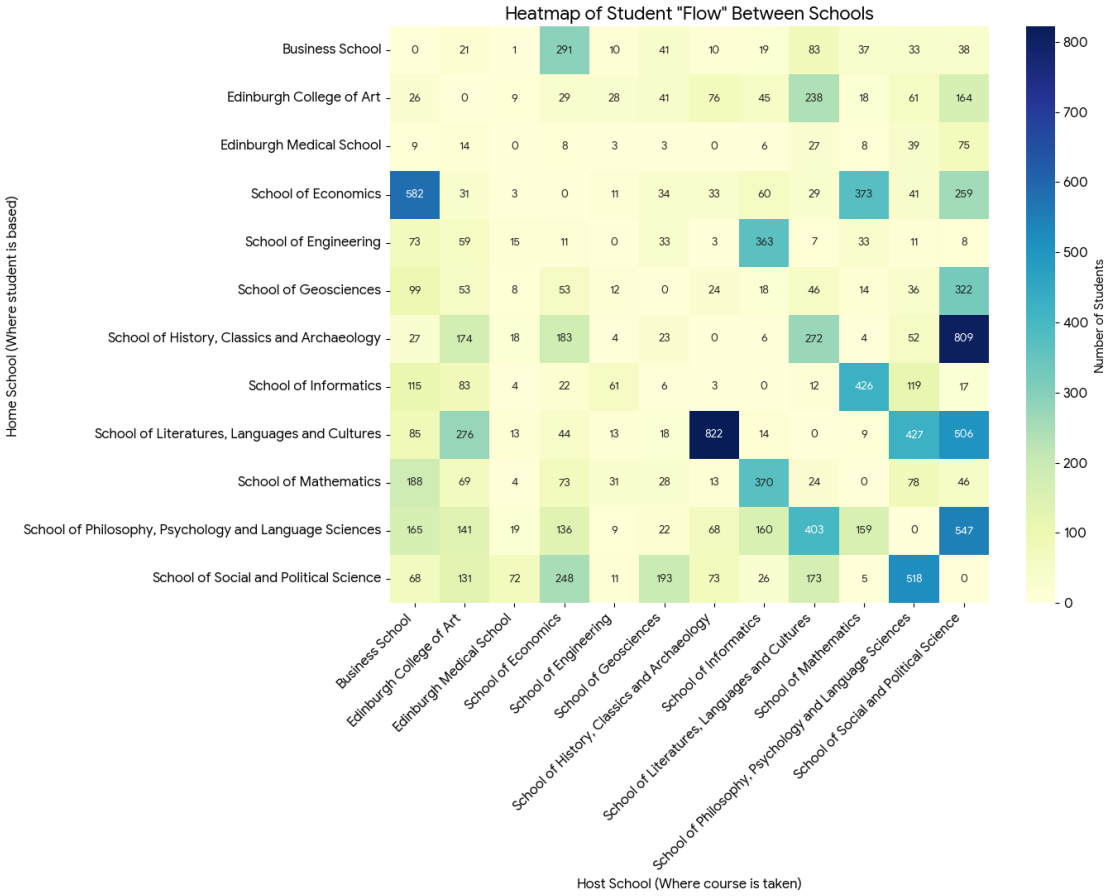


Figure 13: Between School Flow

2. Tracked "Sub" Degrees: This method would require much more support to implement. This would take the existing student choice mechanism for optional courses and instead develop a number of "sub" degree types that students can choose. This would effectively create "tracks" where a student studying an MSc in Statistics and Operational Research would choose between sets of predefined optional course groupings, ideally themed, such as energy, computational optimisation, or logistics. This would greatly reduce the number of possible course combinations and help to narrow down or even partition the possible solution space and avoid further relaxation of constraints. This could also prevent the high external to internal student enrolment ratios seen in the first two years.

3. Teaching Format Rework: This method would require the most effort and buy-in among university staff. This could take many different forms; one such example would be developing an alternating day schedule. Courses are assigned to either "A" or "B" days, lectures and workshops are no longer done in up to 2-hour blocks, but instead all fit the same 50-minute format. Students take the same courses in the same time slots on each "A" and "B" day. Courses could flexibly replace lectures on a given day with workshops. This method would have the same goal to try to reduce course variability and further constrain the possible space. By redesigning the teaching format, it could result in a problem space that behaves better under OR solutions. This would be a significant endeavour and would need experimental timetables to be constructed to confirm viability.

7 Future Challenges and Extensions

7.1 Challenges yet to be Addressed:

Attempts to implement this approach at scale would be complicated by several factors. The first of these is the compute required to achieve an optimal solution. Without caps on runtime or acceptable optimality gap, the model as implemented requires well over 47 hours to reach optimality. This is in part due to the degree to which the model must be relaxed to ensure feasibility across the university, the entirety of which must be considered to properly account for the interactions between modules and students in various schools. These relaxations also reduce solution interpretability, as the objective function captures more metrics which must be balanced against each other in ways that become increasingly opaque with more relaxations.
Both of these contribute to perhaps the greatest challenge facing the model, which is the ease with which it may be implemented by a non-specialist in operational research. Not only does the model require high levels of compute to produce relatively uninterpretable results, it also currently requires knowledge of Python, Anaconda, and Gurobi to implement and adapt to future changes in modules and curricula. All of these factors render the model cumbersome as a tool in its current form and warrant further work.

7.2 Possible Extensions:

In this section we highlight possible extensions that could lead to improvements in the model an its results. The first and potentially the most useful would be a full restructuring of the dataset. A major challenge throughout the project was that the raw data was not immediately suitable for optimisation, with many duplicated events, inconsistent labels, unclear programme structures, and conflicting details. Having a cleaner and more structured dataset would reduce preprocessing demands and make the model easier to scale.

Another extension to implement would be to consider accounting more properly for student choice. The model currently distinguishes between compulsory and optional courses, but does not fully capture the full range of combinations that students might actually take. This was one of the factors contributing to infeasibility in the full-university model. A more detailed treatment of optionality would allow the timetable to better reflect real student pathways and reduce the need for relaxation, as while it might be likely that an economics student may take a business course, it is less likely a mathematics student would take an archaeology course.

A further extension to consider would be to include travel time between different teaching locations. Although the current model accounts for clashes and room usage, it does not consider whether students or staff can move realistically between consecutive events in different buildings or campuses, for example, a student moving from King's Campus to St. Leonard's Land would not be possible for consecutive events. Including travel-time constraints would therefore improve the practical feasibility of the timetable and make the model more representative of the real university setting.

7.3 Alternative Approaches:

A possible alternative would be a decomposition-based hybrid method (Günther R. Raidl [8]). Rather than solving the full university timetable directly, each school could first be optimised independently to produce a set of strong individual school level timetables. These candidate timetables could then be supplied to the Genetic Algorithm as an initial population, allowing the GA to focus on resolving cross-school interactions instead of exploring the entire search space from scratch. The best combined solution could then be passed back into the base MIP model for final refinement. This approach could reduce runtime and improve solution quality by combining local exact optimisation, global heuristic search, and final mathematical polishing, although it could risk losing globally superior solutions if it is the case that the school-level timetables are too restrictive.

8 Conclusion

Motivated by operational research literature on the subject of timetabling, a mixed-integer linear program was formulated and implemented to address timetabling at the University of Edinburgh. The aim was to address questions within the time scenario set of questions proposed by the timetabling team.

Initial testing was carried out on a smaller dataset of only the university's veterinary school which determined that the proposed approach was feasible. The model generated an optimal solution with no clashes and an even spread of events. Following this the model was implemented on the full dataset for all of the courses in the university. Issues with the dataset, such as the Medical School requiring >40 hours of teaching time per week, and the size of the problem space were identified. This required relaxation of the model in order to obtain valid results within the time frame of the project. A final MIP gap of 5.19% was achieved on this model after a 47 hour run which made it clear that the approach would need to be altered to improve efficiency. This lead to the implementation of a genetic algorithm more quickly explore the solution space and aid in the quicker convergence to optimality. This achieved a final gap of 46.81% in 29 hours. This demonstrated the potential of the approach however, time constraints limited the ability to come to a definitive conclusion.

The results from this report lead to the conclusion that the model could be used successfully for this application. However, it requires further fine tuning and greater computational resources to be practical. Furthermore, based on the work outlined in this report a number of non-technical recommendations were laid out. These primarily concern reducing optionality with the system of university programmes and introducing tiered deconfliction meetings, particularly for those schools which demonstrate significant crossover of students with others.

Overall it can be concluded that this approach shows promise as it does generate feasible timetables and with extended run times it does converge towards optimality. However, it is inconclusive on overall practicality and quality of this approach due to requirements for further refinement and compute time. This report therefore recommends further investigation into this methodology.

9 References

[1] Optime Timetabler, EventMap, Avaiable at: https://www.eventmapsolutions.com/products/timetabler, Accessed 11 March 2026

[2] McCollum, B., McMullan, P.J., Parkes, A.J., Burke, E.K., Abdullah S., An Extended Great Deluge Approach to the Examination Timetabling Problem, 2009, 4th Multidisciplinary International Scheduling Conference – Tools & Applications (MISTA)

[3] Lamas Fernandez, C., Matrinez-Sykora, A., Potts, C. (2021), Scheduling Double Round-Robin Sports Tournaments, ITC2021.

[4] Shah, V., Gadhvi, A., Oza, N., Sankhe, M. and Mangla, M. (2025) Optimizing Timetable Generation for Educational Institute Using Genetic Algorithm. RAMSITA 2025

[5] Arias-Osorio, J. and Mora-Esquivel, A. (2020) 'A solution to the university course timetabling problem using a hybrid method based on genetic algorithms', DYNA, 87(215), pp. 47–56.

[6] Alghamdi, H., Alsubait, T., Alhakami, H., Baz, A. (2021), A Review of Optimization Algorithms for University Timetable Scheduling, Engineering, Technology & Applied Science Research, 10(6), Accessed 13 March 2026

[7] Sherali, Hanif D., and Patrick J. Driscoll. "Course Scheduling and Timetabling at USMA." Military Operations Research 4, no. 2 (1999): 25–43.

[8] Günther R. Raidl, Decomposition based hybrid metaheuristics, European Journal of Operational Research, Volume 244, Issue 1, 2015,Pages 66-76

[9] Cambazard, H., Demazeau, F., Jussien, N. and David, P. (2005) Interactively solving school timetabling problems using extensions of constraint programming, in Burke, E. and Trick, M. (eds.) Practice and Theory of Automated Timetabling V (PATAT 2004), Lecture Notes in Computer Science, vol. 3616. Berlin and Heidelberg: Springer, pp. 190–207. https://doi.org/10.1007/11593577$_12$.

[10] Aygül, Ö., Hellgren, T., Azizi, S. and Trapp, A. C. (2026) 'A predict-and-prescribe framework for dynamic course scheduling toward strategic university scaling', Omega, 138, 103406. https://doi.org/10.1016/j.omega.2025.103406

[11] Gurobi Optimization, LLC (2024) Parameter guidelines. Gurobi Optimizer Documentation. Available at: https://docs.gurobi.com/projects/optimizer/en/current/concepts/parameters/guidelines.html (Accessed: 24 March 2026).

10 AI Disclosure

AI was used as a supplementary tool in this project. Primarily for the identification of errors in code scripts and brainstorming of adjustments and their practicalities.

11 Appendix

11.1 Mathematical Formulation

11.1.1 Sets

First of all, we want to define the sets of the problem in order to be able to distinguish all kinds of events. The first ones that we need to define are:

- K: Set of all 1-year long programs or 1 year from a longer programme
- M: Set of all distinct events a course can have with first item representing lectures

Now, let $I_{k_1}, I_{k_2}, I_{k_3}, \ldots$ denote families of 1-year long programs or 1 year from a longer programme. If two courses belong in the same set I_k, then they are part of the same year of the respective programme. Let's also define A_k and B_k, such that

$$I_k = A_k \cup B_k$$

where:

- A_k is the set of compulsory courses,
- B_k is the set of optional courses.

Similarly, there are roughly 100 types of events (e.g. lecture, workshop, tutorial, presentation, exam etc.) which we will also distinguish using the same method. We define

- A_k^m as the set of event type m of compulsory courses,
- B_k^m as the set of event type m of optional courses.

Obviously,

$$\bigcup_{m \in M} A_k^m = A_k \quad \text{and} \quad \bigcup_{m \in M} B_k^m = B_k$$

If an event is indexed as an all class event in the event calendar, then it becomes part of A_k by default even if it is not an actual course, indicating that all students in the respective programme must attend in any case.

As for the time and space definitions, the following sets will be used:

- $T = \{1, 2, ..., 9\}$: Set of teaching hours during a day
- $D = \{1, 2, 3, 4, 5\}$: Set of teaching days in a week
- C: Set of room capacities

11.1.2 Parameters and Variables

The parameters of the problem are the following:

- $d_{i,m}$: hours needed for event m of course i per week
- P_k: total number of courses available in the year/programme k
- P'_k: total number of courses a student needs to pass in the year/programme k in order to progress
- $s_{i,m}$: Number of seats allocated to event m of course i
- r_c : Number of rooms with capacity c

- W: Potential weight for the objective function

Having said that, the decision variables are the following:

- $x_{i,m,t,d} = \begin{cases} 1, & \text{if course } i \text{ event } m \text{ is scheduled at time slot } t \text{ of day d,} \\ 0, & \text{otherwise.} \end{cases}$
- y_k: slack variable, designed to penalize optional course overlaps in year/programme k
- b_d: slack variable, designed to penalize the absence of a lunch break in day d

11.1.3 Constraints

Core Constraints: Now we want to define the constraints of our formulation.
First of all, we don't want any overlapping lectures for compulsory courses within the same programme/year:

$$\sum_{i \in A_k} x_{i,1,t,d} \leq 1 \quad \forall k \in K,\ t \in T,\ d \in D \quad (1)$$

It is also important to consider that if we are unable to deter any overlapping between optional courses, we should minimize it as much as possible, hence:

$$\sum_{i \in B_k, m \in M} x_{i,mt,d} \leq y_k \quad \forall k \in K,\ t \in T,\ d \in D \quad (2)$$

Also, each event has a duration of $d_{i,m}$ hours per week that shall be determined:

$$\sum_{d \in D} \sum_{t \in T} \sum_{i \in A_k^m} x_{i,m,t,d} = d_{i,m} \quad \forall k \in K, m \in M \quad (3)$$

Finally, in order to make sure that the existing timetable can be implemented based on the available university venues, we want the courses that take place simultaneously to fit in the rooms of the existing facilities:

$$\sum_{k \in K} \sum_{\substack{i \in I_k,\ m \in M: \\ s_{i,m} \leq c}} x_{i,t,d} \leq \sum_{c' \leq c} r_{c'} \quad \forall t \in T,\ d \in D,\ c \in C \quad (4)$$

Questions Constraints: In this section we want to make an endeavor to answer any questions regarding the time management of the timetable. These questions are the following:

1. Would it be possible to reduce core teaching hours to Monday-Friday 9-5pm?

$$x_{i,9,d} = 0 \quad \forall i \in I_k,\ k \in K,\ d \in D \quad (5)$$

2. Would it be possible to reduce core teaching hours to eliminate teaching on 12pm-6pm on Friday, with Monday-Thursday remaining the same?

$$x_{i,4,5} = x_{i,5,5} = x_{i,6,5} = x_{i,7,5} = x_{i,8,5} = x_{i,9,5} = 0 \quad \forall i \in I_k,\ k \in K \quad (6)$$

3. Could core teaching/lectures still be delivered without clashes?
(We assume that $m = 1$ represents lecture events)

$$\sum_{i \in A_k^1 \cup B_k^1} x_{i,t,d} \leq 1 \quad \forall k \in K,\ t \in T,\ d \in D \quad (7)$$

4. Could you provide a lunch break for the majority of students between 12-2pm in each scenario as well?

$$\sum_{i \in I_k} (x_{i,4,d} + x_{i,5,d}) - 1 \leq b_d \quad \forall k \in K,\ d \in D$$

$$b_d \geq 0 \quad \forall d \in D \quad (8)$$

Extension Constraints: Another thing that we should also consider is to make the timetable likeable by both the students and the teaching staff, independent of its feasibility.

Our first consideration is that it might be more convenient to have consecutive events for the same course, for example 2-hour lectures. The generalization of it for n consecutive hours can be formulated as:

$$x_{i,t,d} \leq x_{i,t+1,d} + x_{i,t-1,d}$$

$$2x_{i,t,d} \leq x_{i,t-2,d} + x_{i,t-1,d} + x_{i,t+1,d} + x_{i,t+2,d}$$

$$\vdots$$

$$nx_{i,t,d} \leq \sum_{r=1}^{n} (x_{i,t-r,d} + x_{i,t+r,d})$$

$$\forall i \in I_k,\ k \in K,\ t \in \{2, \ldots, 8\},\ d \in D \quad (9)$$

It should be noted that the $\forall i \in I_k$ can be violated depending on the preferences of each programme.

What we also want to consider is that the daily teaching load is less than 6 hours so students are able to fully concentrate during the whole day, thus:

$$\sum_{t \in T} \sum_{i \in I_k} x_{i,t,d} \leq 6 \quad \forall k \in K,\ d \in D \quad (10)$$

11.1.4 Objective

Finally, we want to minimize the total overlaps occurring in all the programs, and also penalize the absence of a lunch break (if we are considering that). Hence, the objective function is given by:

$$\min \sum_{k \in K} y_k + W \sum_{d \in D} b_d$$

$$\text{Subject to:} \quad (1) - (4)$$

11.2 Relaxation

To apply the model to the entire university's dataset, it was necessary to relax the constraints related to the questions we were asked. In the strictest sense, this means that the answer to all questions is no. Instead, this formulation is intended to determine the degree to which the objectives mentioned in timetabling's questions can be achieved. This can be determined using the following formulation using the same parameters and sets, with new set R the set of rules and new decision variables l_{dk} the number of hours over 6 scheduled on d for k q_{rk} the number of violations of rule r by programme

k.

$$\min \quad \sum_{k \in K} \left(W y_k + \sum_{r \in R} q_{rk} \right) \tag{1}$$

$$\text{s.t.} \quad (1) - (8), (10) \tag{2}$$

$$\sum_{i \in I_k} \sum_{m \in M} \sum_{d \in D} x_{i,m,9,d} \leq q_{1k} \qquad \forall k \in K \tag{3}$$

$$\sum_{i \in I_k} \sum_{m \in M} \sum_{t \geq 4} x_{i,m,t,5} \leq q_{2k} \qquad \forall k \in K \tag{4}$$

$$\sum_{d \in D} b_{dk} \leq q_{3k} \qquad \forall k \in K \tag{5}$$

$$\sum_{i \in I_k} \sum_{m \in M} \sum_{t \in T} x_{i,m,t,5} \leq 6 + l_{dk} \qquad \forall d \in D, \forall k \in K \tag{6}$$

$$\sum_{d \in D} l_{dk} \leq q_{4k} \qquad \forall k \in K \tag{7}$$

This minimizes both the number of clashes within a programme across all programs and the number of rule violations. The weighting of these two objectives naturally plays a role in the solution obtained and can be tuned according to preference. In this case, $W = 10$ was used to represent an assumed preference for enabling students to choose from as many optionals as offered within their programme as possible over the various time conveniences considered.