10th International Conference of the Practice and Theory of Automated Timetabling
PATAT 2014, 26-29 August 2014, York, United Kingdom

Room Allocation Optimisation at the Technical University of Denmark

Niels-Christian Fink Bagger · Jesper Larsen · Thomas Stidsen

Keywords University Course Timetabling · Room Allocation · Mathematical Programming · Heuristics · Matheuristics

1 Introduction

As at many other universities the Technical University of Denmark (DTU) faces the challenge of solving a case of the curriculum based university course timetabling problem (CUCTT) multiple times a year. However, there are some slight modifications to the CUCTT problem usually described in the literature. One of the major difference is that the assignment of the courses to specific time slots are predetermined and cannot be subject to changes. This is a decision made by the administration since this takes away the issue of course collisions, e.g. when two courses sharing a student are allocated at overlapping time slots, since the students are to ensure by themselves that their courses do not overlap. The problem was first considered in the masters' thesis [1] and the project here is an extension of the work done in that thesis.

2 Objectives

For each course a predetermined set of events will occur during the semester, e.g. lectures, laboratory exercises, tutorials and so on. Each event may be allocated to more than one room. It is allowed to allocate the events such that the sum of the capacities of the allocated rooms does not accommodate all the students attending the event and an objective is to minimize this number of unallocated students. Some of the lecturers prefers to give their lectures

Niels-Christian Fink Bagger, Jesper Larsen, Thomas Stidsen
Technical University of Denmark
Tel.: +45 45 25 25 42
E-mail: nbag@dtu.dk

in specific rooms. An objective is to minimise the geographical distance between the allocated rooms and the requested rooms of the events. A highly prioritised soft constraint is that events from the same course occurring the same day must be allocated as close to each other as possible with respect to their geographical location and if an event is split into multiple rooms, these must also be as close to each other as possible in the geographical sense. The difference between this soft constraint and the *Room Stability* considered in the curriculum timetabling problem of ITC2007 [2] is that in ITC2007 the number of different rooms that a course or event is assigned to is penalised whereas here the actual physical distance between the rooms are penalised. As an example of the difference between the two penalties consider Fig. 1.

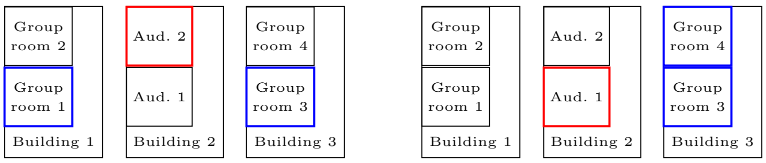

Fig. 1 Example of a course consisting of two events; a lecture followed by group exercises. The auditorium coloured in red is the selected room for the lecture and the two group rooms marked in blue are the selected rooms for the group exercises. The two have the same *Room Stability* penalty, however the solution to the right is considered to be better than the solution to the left since the allocated rooms are closer to each other.

3 Solution Approach

The Mixed Integer Program (MIP) formulation describing the problem is given in Fig. 2.

The following sets and parameters are given as input to the model:

- E, R, C are sets of events, rooms and courses respectively
- $\mathcal{C}$ is every distinct pair of events which are from the same course and occurs on the same day of the week
- $\mathcal{R}$ is every distinct pair of rooms
- χ is every distinct pair of events which are overlapping in time but not from the same course.
- P_r is the capacity of room $r \in R$
- $F_{e,r}$ is 1 if event $e \in E$ is allowed to be scheduled in room $r \in R$ and 0 otherwise
- $R_e^{\max}$ is the maximum number of rooms that event $e \in E$ is allowed to be split into.
- The parameters α_e, $\gamma_{e,r}$, $\zeta^e_{r,r'}$ and $\zeta^{e,e'}_{r,r'}$ for penalising respectively, the number of unseated students of event $e \in E$, the allocation of event $e \in E$ to room $r \in R$, the allocation of event $e \in E$ to both room $r \in R$ and $r' \in R$ and the allocation of events $e \in E$ and $e' \in E$ to rooms $r \in R$ and $r' \in R$

The binary variable $x_{e,r}$ takes value 1 if event $e \in E$ is scheduled in course $r \in R$ and 0 otherwise. The variable y_e will take the value of the number of students from event $e \in E$ that are not seated in the solution. The variable $t^e_{r,r'}$ will take value 1 if event $e \in E$ is scheduled in both room $r \in R$ and in room $r' \in R$. The variable $u^{e,e'}_{r,r'}$ will take value 1 if event $e \in E$ is scheduled in room $r \in R$ and event $e' \in E$ is scheduled in room $r' \in R$.

$$
\begin{array}{llll}
\min & \displaystyle\sum_{e \in E} \alpha_e \cdot y_e + \sum_{e \in E, r \in R} \gamma_{e,r} \cdot x_{e,r} + \sum_{\substack{e \in E,\\ (r,r') \in \mathcal{R}}} \zeta^e_{r,r'} \cdot t^e_{r,r'} + \sum_{\substack{(e,e)' \in \mathcal{C}\\ (r,r') \in \mathcal{R}}} \zeta^{e,e'}_{r,r'} \cdot u^{e,e'}_{r,r'} & & (1)\\
\text{s.t.} & x_{e,r} \leq F_{e,r} & \forall e \in E, r \in R & (2)\\
& \displaystyle\sum_{r \in R} x_{e,r} \leq R^{\max}_e & \forall e \in E & (3)\\
& \displaystyle\sum_{r \in R} P_r \cdot x_{e,r} + y_e \geq S_e & \forall e \in E & (4)\\
& x_{e,r} + x_{e',r} \leq 1 & \forall (e,e') \in \chi, r \in R & (5)\\
& x_{e,r} + x_{e,r'} - t^e_{r,r'} \geq 1 & \forall e \in E, (r,r') \in \mathcal{R} & (6)\\
& x_{e,r} + x_{e',r'} - u^{e,e'}_{r,r'} \leq 1 & \forall (e,e') \in \mathcal{C}, (r,r') \in \mathcal{R} & (7)\\
& x_{e,r} \in \mathbb{B} & \forall e \in E, r \in R & \\
& y_e \geq 0 & \forall e \in E & \\
& t^e_{r,r'} \geq 0 & \forall e \in E, (r,r') \in \mathcal{R} & \\
& u^{e,e'}_{r,r'} \geq 0 & \forall (e,e') \in \mathcal{C}, (r,r') \in \mathcal{R} &
\end{array}
$$

Fig. 2 Model of the room allocation problem.

The description of the objective and the constraints follows:

(1) The weighted sum of all the soft constraints
(2) Event $e \in E$ can only be scheduled into room $r \in R$ if it is marked as feasible
(3) Event $e \in E$ can at most be put into $R^{\max}_e$ rooms
(4) If event $e \in E$ is put into rooms where the total capacity is less than the number of students S_e then y_e is given a lower bound of the difference, i.e. the number of students which are unallocated
(5) Two events, $e \in E$ and $e' \in E$, which are overlapping cannot be scheduled in the same time slot
(6) If an event $e \in E$ is allocated to both room $r \in R$ and $r' \in R$ then $t^e_{r,r'}$ is given a lower bound of 1
(7) If event $e \in E$ is allocated to room $r \in R$ and event $e' \in E$ is allocated to room $r' \in R$ then $u^{e,e'}_{r,r'}$ is given a lower bound of 1

3.1 Greedy Graph Colouring Constructive (Mat)Heuristic

Since the problem has similarities to graph colouring and the k–clique problem it seems natural to implement algorithms inspired from solution approaches to these problems. This is for instance done for the examination timetabling problem in [4] showing good results. The idea of ordering events due to "difficulty" is adopted.

The heuristic works by iteratively making a lexicographic order of the unscheduled events by "difficulties" and preferences and then picking the best allocation for the first event in the list. The "difficulty" of an event is based on the hard constraints of the mathematical model that the event is part of, i.e. how difficult it is expected to be to schedule. Since different planners have different priorities of the preferences the heuristic will adapt the choice of the ordering during the search. This is done by assigning a score $s_{i,j}$ for each pair of soft or hard constraints i and j taking an initial value of 1 indicating how well the algorithm performs when the events are ordered lexicographically by i before j and then updated in each iteration as a result of the performance. The ordering of the constraints is done by the algorithm described in Algorithm 1.

Algorithm 1: OrderList

```
  Input: An unordered list of the indces of the constraints $L$
  Output: An ordered list $O$
1 Pick the first element in $L$, remove it from $L$ and insert it in $O$
2 while $L \neq \emptyset$ do
3     Pick the first element $i$ in $L$ and remove it from $L$
4     foreach $j \in O$ in the given order do
5         Pick a random number $r \in [0, 1]$
6         if $r \leq \frac{s_{i,j}}{s_{i,j}+s_{j,i}}$ then
7             Insert $i$ in $O$ right before $j$
8             Exit the foreach-loop
9     If $i$ did not get inserted then insert $i$ last in $O$
```

After the constraints have been ordered the events are put into a sorted list in the given lexicographical order. Then the first event from the list is taken out and allocated to a room which decreases the objective value. If such a room cannot be found then the next event in the list is chosen. The algorithm stops the first time an event is allocated a room and then the constraint list is reordered for the next iteration of the heuristic.

This heuristic has also been extended into a matheuristic. The basic idea is the same but instead of considering only one event of the time the matheuristic chooses the first k events in the ordered list. Then the given MIP model is solved to find the best allocation for the k events where previously allocated events are fixed at their location and unallocated events, which are not one of the k chosen events, are ignored.

4 Results

For comparison the heuristic and the matheuristic has been implemented in C# an tested on five real-life data sets from DTU. The model has also been tested by a direct implementation in Gurobi[3] v. 5.5.1. The time limit has been set to fifteen minutes and an Overview of the results can be seen in Table 1.

	H		MH		GRB		
Name	Time	UB	Time	UB	Time	UB	LB
DataSet1	6	4016031	91	1408853	900	1505480	1155852
DataSet2	7	7057059	109	2857753	900	3055835	1668848
DataSet3	14	3290617	153	2039970	900	1938341	1500971
DataSet4	5	8083225	90	7254350	900	6629019	5934471
DataSet5	3	4193801	64	1582210	900	1525580	1451168
Average	7	5328147	101	3028627	900	2930851	2342262

Table 1 Comparison of the heuristic (H), matheuristic (MH) and Gurobi (GRB). UB is the obtained value, Time is the time that the algorithm spent in seconds, LB is the lower bound.

From Table 1 it can be seen that the heuristic is outperformed by both the matheuristic and Gurobi in terms of the obtained solution. The matheuristic and Gurobi are very close, however the matheuristic obtains the solutions within a much smaller amount of time and since it is a constructive heuristic it can potentially get a better solution if some improvement step is implemented. The use of a MIP solver inside the constructive heuristic is an easy way to extend the heuristic and in this case improves the performance significantly. However the project is still in a very preliminary state and other heuristics known for performing well from the literature needs to be implemented for comparison.

References

1. Bærentsen, R.: Optimization of room-allocation at the technical university of denmark. Masters' thesis, Technical University of Denmark
2. Gasparo, L.d., McCollum, B., Schaerf, A.: The second international timetabling competition (itc-2007): Curriculum-based course timetabling (track 3). Tech. rep. (2007). Http://www.cs.qub.ac.uk/itc2007/curriculmcourse/report/curriculumtechreport.pdf
3. Gurobi Optimization, Inc.: Gurobi Optimizer Reference Manual (2014). URL http://www.gurobi.com
4. Sabar, N.R., Ayob, M., Qu, R., Kendall, G.: A graph coloring constructive hyper-heuristic for examination timetabling problems. Applied Intelligence **37**(1), 1–11 (2012)