European Journal of Operational Research 266 (2018) 35–45

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Discrete Optimization

A strategic view of University timetabling

Michael Lindahl[a,c,*], Andrew J. Mason[b], Thomas Stidsen[a], Matias Sørensen[c]

[a] Department of Management Engineering, Technical University of Denmark, Produktionstorvet 426, 2800 Lyngby, Denmark
[b] Engineering Science, University of Auckland, 70 Symonds St, Auckland 1010, New Zealand
[c] MaCom A/S, Vesterbrogade 48, 1620 Copenhagen, Denmark

ARTICLE INFO

Article history:
Received 23 December 2016
Accepted 14 September 2017
Available online 22 September 2017

Keywords:
Timetabling
Multiple objective programming
Integer programming

ABSTRACT

University timetabling has traditionally been studied as an operational problem where the goal is to assign lectures to rooms and timeslots and create timetables of high quality for students and teachers. Two other important decision problems arise before this can be solved: what rooms are necessary, and in which teaching periods? These decisions may have a large impact on the resulting timetables and are rarely changed or even discussed. This paper focuses on solving these two strategic problems and investigates the impact of these decisions on the quality of the resulting timetables.

The relationship and differences between operational, tactical and strategic timetabling problems are reviewed. Based on the formulation of curriculum-based course timetabling and data from the Second International Timetabling Competition (ITC 2007), three new bi-objective mixed-integer models are formulated. We propose an algorithm based on the ϵ-constraint method to solve them. The algorithm can be used to analyze the impact of having different resources available on most timetabling problems. Finally, we report results on how the three objectives – rooms, teaching periods, and quality – influence one another.

1. Introduction

Educational timetabling has gained a lot of attention within operations research and has been studied for a long time; for an overview see Kingston (2013) and Kristiansen and Stidsen (2013). Many variations of the problem exist as it varies between different educational stages and countries. A definition commonly used for timetabling comes from Wren (1996):

"Timetabling is the allocation, subject to constraints of given resources to objects being placed in space time, in such a way as to satisfy as nearly as possible a set of desirable objectives."

This definition captures the timetabling problems that are usually investigated in the literature. However, other important related decisions are rarely investigated. For example: Which resources should be available? Where in space and time is it allowed to place these objects? In this paper, we create models and methods to decide what room sizes and teaching periods should be used by an organization, and analyzes how these decisions affect the resulting timetable.

An overview of the decision problems that relate to timetabling are illustrated in Fig. 1. The value of a timetable obtained by solving these problems is characterized by three measures associated with the timetabling process:

Agility. Responding to changes and quickly make alterations to the timetable.
Quality. A timetable that gives a good work environment for employees and students.
Cost. A timetable which uses costly resources efficiently.

We categorize each of these problems as either operational, tactical or strategic. The strategic problems are the ones that are solved first and which then affect the tactical problems that are solved next. Finally, the operational problems are solved last. The problems we find at each level, as shown in Fig. 1, are as follows:

Operational. The operational problems are the ones that result in final timetable, i.e. deciding which lectures should be taught in what rooms and when. This main problem is solved every semester, and we separate it into two closely related cases. The first case is the *assignment problem*, which is the feasibility problem of finding a conflict-free timetable.

A common addition to this problem, which is the second case, is including quality measures that aim to make a desirable timetable for students and staff, i.e. the *Quality problem*. Soft constraints are used in this case, where the violation of these should be minimized. Both course timetabling problems for the Second International Timetabling Competition, formulated in Di Gaspero, McCollum, and Schaerf (2007), fall into this second category.

* Corresponding author at: Department of Management Engineering, Technical University of Denmark, Denmark.
E-mail addresses: miclin@dtu.dk, miclindahl@gmail.com (M. Lindahl), a.mason@auckland.ac.nz (A.J. Mason), thst@dtu.dk (T. Stidsen), ms@macom.dk (M. Sørensen).

https://doi.org/10.1016/j.ejor.2017.09.022
0377-2217/

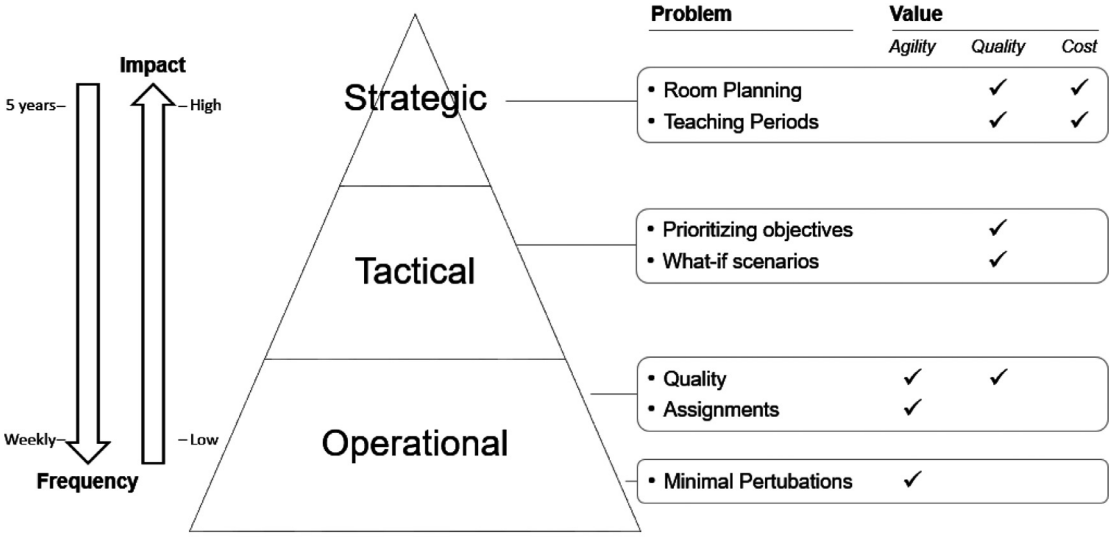

Fig. 1. Different variations of the timetabling problem occur at different strategic levels. Each problem contributes with different value to the organization when solved. The top problems are long-term decisions that have an impact on all timetables whereas the problems at the bottom have lower impact but are decisions that need to be taken often.

After a timetable is put into production, disruptions will often occur during the semester as alterations need to be made. Handling such disruptions requires solving the minimal perturbation problems addressed in Muller, Rudová, and Barták (2005) and Phillips, Walker, Ehrgott, and Ryan (2014). When decision-support tools are used to solve these problems, it provides agility by being able to create the timetable more quickly and respond faster to changes.

Tactical. The tactical problems involve the decisions that are taken before creating the actual assignments. This includes, for example, to prioritize the different quality measures or how to divide the timetabling workload between multiple planners. These decisions affect the quality because a course could end up being taught in an undesired timeslot because the planner does not have other rooms available, even though there may be free rooms in another part of campus.

Strategic. The strategic problems are the long-term decisions that affect the timetable and have a high impact on the organization. This could, for example, be to allow teaching to be scheduled later in the day by adding additional timeslots, which we define as the *Teaching Periods* problem. Adding extra timeslots has a high impact on the timetable for professors and students and would therefore often be a decision made by the top management. Buildings and rooms are high-cost areas for universities because new buildings are expensive and unused rooms can often be turned into offices or rented out. However, these decisions can not be changed from year to year, and it is, therefore, crucial that new rooms are of the actually needed size, and that a room is not rented out for five years if it is required in the next semester. We refer to this problem as the *Room Planning* problem. The authors have experienced great interest in these strategic problems from Aarhus University, Technical University of Denmark and Roskilde University.

Compared to the operational and tactical problems, strategic problems have received little attention in the literature. The *Room Planning* problem was first explored in Fizzano and Swanson (2000), where they show how their assignment model can be used to find the minimum number of rooms necessary to create a feasible schedule. This is done iteratively by removing rooms until the problem becomes infeasible. Beyrouthy et al. (2007) analyze how available rooms affect the utilization and visualize this in multiple ways. They then show how the room profile can be adjusted to maximize utilization. They investigate room robustness by sampling the sizes of the courses under different scenarios. They do not, however, include timetable constraints such as curricula or teachers that can make it impossible to create a timetable that does not conflict with the given rooms.

Beyrouthy et al. (2009) include timetabling constraints and examine the utilization of rooms by determining how many courses can be planned with a given room profile. They conclude that different constraints contribute to lower utilization. Beyrouthy, Burke, McCollum, McMullan, and Parkes (2010) investigate space type planning, where space types refer to various kinds of rooms such as lecture halls and tutorial rooms. The total capacity is fixed. They do not look into how the room profile affects the quality of the timetable for students.

The *Teaching Periods* problem of choosing the number of timeslots has not previously been investigated. Decision-support tools for management that support these important decisions are limited, or rather non-existent.

The purpose of this paper is to create new insights into how these strategic decisions affect the timetable and to lay the foundation for future research within this area. We will investigate the two strategic decisions of deciding which rooms to use and how many timeslots there should be. We will also analyze how these decisions affect the quality of the timetable by using bi-objective optimization and mixed-integer programming.

In Section 2 we define the strategic problems and in Section 3 we show the method used to solve these problems. The results are shown in Section 4, and finally conclusions and suggestions for future research are given in Section 5.

2. Curriculum-based course timetabling

University timetabling differs a lot between universities due to differences between educational traditions in different countries. Therefore, it has been difficult for researchers to compare their work, as they were solving different problems. To overcome this problem, timetabling competitions have been held where common problem formulations and benchmark instances have been formulated.

As a basis for this analysis, we use the data-sets and problem formulation of Curriculum-based Course Timetabling (CB-CTT) from the Second International Timetabling Competition ITC-2007

stated in Di Gaspero et al. (2007). For an overview of the research done on this problem, we recommend (Bettinelli, Cacchiani, Roberti, & Toth, 2015). As discussed, this is an operational problem, and we will now give the formulation of the original problem and then show how it can be used as a basis to solve the strategic problems.

The original curriculum-based course timetabling problem is formulated as follows: A set of courses is given, and each course consists of a number of lectures that should be planned. A timeslot and a room should be assigned to each lecture without causing conflict, where a timeslot is a time on a particular weekday. Two lectures from one course cannot take place in the same timeslot, as this will cause a conflict. Besides courses and timeslots, a list of rooms is also given, and only one lecture can be taught in a room in one timeslot. Each room also has a certain size, and it is a requirement that a room should be able to accommodate all the students following a given course allocated in that room. Each course is associated with a teacher, and a teacher can only teach one course at the time. Finally, we also have a set of curricula that consist of a set of courses that cannot be placed in the same timeslot.

The original formulation includes four different quality measures (soft-constraints): `RoomCapacity`, stating that the room should be able to accommodate all students; `RoomStability` that states that all lectures from the same course should be planned in the same room; `MinimumWorkingDays` ensures that a lecture is spread over a minimum number of days specified in the data, where a penalty is paid for each day not used, and finally, `CurriculumCompactness`, where a penalty is paid if a lecture from a curriculum is not scheduled next to a course from the same curriculum. To create our models, we define the following sets:

$\mathcal{C}$: set of courses
$\mathcal{CU}$: set of curricula
$\mathcal{P}$: set of timeslots across the week
$\mathcal{D} = \{P_{\text{Mo}}, P_{\text{Tu}}, P_{\text{We}}, P_{\text{Th}}, P_{\text{Fr}}\}$, set of timeslots belonging to each day of the week e.g. P_{Mo} is all the timeslots on Mondays.
$\mathcal{R}$: set of rooms

We then have the following parameters:

$l(c)$: the number of lectures for course $c \in \mathcal{C}$
$mnd(c)$: the minimum number of weekdays on which there must be a lecture for course $c \in \mathcal{C}$
$dem(c)$: the demand for course $c \in \mathcal{C}$, i.e. the number of students in that course.
$cap(r)$: the capacity of room $r \in \mathcal{R}$

We also define helper sets that are used to formulate the model:

$\mathcal{S} = \{cap(r) : r \in \mathcal{R}\} \cup \{0\}$, set of unique room capacities, including zero
$\mathcal{S}_{\geq s} = \{s' \in \mathcal{S} : s' \geq s\}$, set of room capacities larger than or equal to $s \in \mathcal{S}$
$\mathcal{C}_{\geq s} = \{c \in \mathcal{C} : dem(c) \geq s\}$, set of courses with a demand larger than or equal to $s \in \mathcal{S}$
$\mathcal{R}_{\geq s} = \{r \in \mathcal{R} : cap(r) \geq s\}$, set of rooms with a capacity larger than or equal to $s \in \mathcal{S}$

2.1. Quality problem

The Quality problem is the standard problem in timetabling where the goal is to create a feasible timetable of high quality. A number of soft constraints are defined, and the quality is measured by the number of violations of these. Compared to the original description of CB-CCT we make two changes. Because we want to explore the trade-off between the number of rooms, we turn `RoomCapacity` into a hard constraint, meaning that rooms cannot be overbooked with more students than there can be seated.

$$\min \quad f_{qual} = \sum_{c \in \mathcal{C}} 5 \cdot w_c + \sum_{cu \in \mathcal{CU}, p \in \mathcal{P}} 2 \cdot v_{cu,p} \tag{1a}$$

$$\text{s.t.} \quad \sum_{c \in \mathcal{C}_{\geq s}} x_{c,p} \leq |\mathcal{R}_{\geq s}| \qquad \forall s \in \mathcal{S}, p \in \mathcal{P} \tag{1b}$$

$$\sum_{p \in \mathcal{P}} x_{c,p} = L(c) \qquad \forall c \in \mathcal{C} \tag{1c}$$

$$\sum_{p \in d} x_{c,p} - z_{c,d} \geq 0 \qquad \forall c \in \mathcal{C}, d \in \mathcal{D} \tag{1d}$$

$$\sum_{d \in \mathcal{D}} z_{c,d} + w_c \geq mnd(c) \qquad \forall c \in \mathcal{C} \tag{1e}$$

$$\sum_{c \in \mathcal{CU}} x_{c,p} - q_{cu,p} = 0 \qquad \forall cu \in \mathcal{CU}, p \in \mathcal{P} \tag{1f}$$

$$-q_{cu,p-1} + q_{cu,p} - q_{cu,p+1} - v_{cu,p} \leq 0 \qquad \forall cu \in \mathcal{CU}, p \in \mathcal{P} \tag{1g}$$

$$\sum_{c \in \mathcal{C}(t)} x_{c,p} \leq 1 \qquad \forall t \in \mathcal{T}, p \in \mathcal{P} \tag{1h}$$

$$x_{c,p} \in \mathbb{B} \qquad \forall c \in \mathcal{C}, p \in \mathcal{P} \tag{1i}$$

$$w_c \in \mathbb{R}^+ \qquad \forall c \in \mathcal{C} \tag{1j}$$

$$z_{c,d} \in [0,1] \qquad \forall c \in \mathcal{C}, d \in \mathcal{P}_d \tag{1k}$$

$$q_{cu,p} \in [0,1] \qquad \forall cu \in \mathcal{CU}, p \in \mathcal{P} \tag{1l}$$

$$v_{cu,p} \in [0,1] \qquad \forall cu \in \mathcal{CU}, p \in \mathcal{P} \tag{1m}$$

Model 1. The MIP model for the Quality problem.

The second change is that the soft objective `RoomStability` is removed. This is to reduce the computational complexity, because, as shown in Lach and Lübbecke (2012), preventing lectures from being assigned to specific rooms can reduce the number of decision variables significantly. The penalty for `RoomStability` is also the smallest of the soft constraints and contributes the least to the objective. Note that this simplification does not alter the general applicability of our method.

The mixed integer model we use is based on the one proposed in Lach and Lübbecke (2012) and is seen in Model 1. It includes the following variables and constraints: The decision variable is the binary $x_{c,p}$ that indicates if course $c \in \mathcal{C}$ is planned in timeslot $p \in \mathcal{P}$. Constraint (1c) ensures that all lectures are planned. Constraint (1b) ensures that the rooms are able to accommodate all students in each course. Constraint (1h) ensures that a teacher has only one course in a timeslot.

The two objective terms in (1a) are calculated the following way: To calculate `MinimumWorkingDays` we introduce a positive variable $z_{c,d}$ which takes the value 1 if course $c \in \mathcal{C}$ is planned on day $d \in \mathcal{D}$ and otherwise zero. This is ensured by constraint (1d) and the objective pressure that pushes $z_{c,d}$ to one, if possible. The violation is then calculated by constraint (1e) and the variable w_c. To calculate `CurriculumCompactness` we introduce the positive variable $q_{cu,p}$, which takes the value 1 if curriculum $cu \in \mathcal{CU}$ is planned in timeslot $p \in \mathcal{P}$ and zero otherwise; this is ensured by constraint (1f). Constraint (1g) then ensures that the variable $v_{cu,p}$ then takes the value 1 if there is a violation in curriculum $cu \in \mathcal{CU}$ in timeslot $p \in \mathcal{P}$.

2.2. Room Planning problem

The second problem we will consider is the *Room Planning problem* where the objective is to find the minimum number of rooms to use to accommodate all courses to be taught. In this section we will introduce the simplest form of this problem, as given in Model 2.

$$\min \quad f_{seats} = \sum_{s \in \mathcal{S}} s \cdot r_s \tag{2a}$$

$$\text{s.t.} \quad |\mathcal{P}| \sum_{s \in \mathcal{S}_{\geq s}} r_s \geq \sum_{c \in \mathcal{C}_{\geq s}} l(c) \qquad \forall s \in \mathcal{S} \tag{2b}$$

$$r_s \in \mathbb{Z}^+ \qquad \forall s \in \mathcal{S} \tag{2c}$$

Model 2. Set covering MIP-Model for the Room Planning problem.

Instead of having a fixed set of rooms we introduce a decision variable $r_s \in \mathbb{Z}^+$ that determines the number of rooms of size s that should be used. The capacity of a room is correlated with the physical size, and therefore, also with the cost of a room. Thus, we will seek to minimize the total number of seats as defined in Eq. (2a).

A commonly used metric is the utilization defined in Beyrouthy et al. (2007). This utilization calculates the proportion of time, on average, that a seat is occupied. It should be noted that minimizing the number of seats will maximize the utilization.

$$\text{Utilization} = \frac{\sum_{c\in\mathcal{C}} l(c)\cdot dem(c)}{|\mathcal{P}|\sum_{s\in\mathcal{S}} s\cdot r_s}$$

We need to decide which room sizes the model should be able to choose from, i.e. sizes included in the set S. In an optimal room profile, a room will always have the same size as the largest course assigned to it, otherwise it would be wasted capacity. Therefore S should include the sizes of all courses. Because the number of variables, and thus the size of the solution space, depends on the size of S, it is desirable to keep this set small. In a practical setting it is unlikely that a manager would distinguish between rooms of 22 and 25 seats. Consequently, to make the set smaller, we define a parameter δ and only use room sizes that are multiples of the value of this parameter. The largest room will have the size of the largest course rounded up to the nearest multiple of δ. The set S can subsequently be generated in the two following ways:

$$S_{\text{optimal}} = \{dem(c) : c \in \mathcal{C}\}$$

$$S_{\text{approx.}} = \left\{\delta \cdot \left\lceil \frac{dem(c)}{\delta} \right\rceil : c \in \mathcal{C}\right\}$$

When we decide to use a room, we can plan as many lectures in it as we have timeslots ($|\mathcal{P}|$). To have enough rooms to plan all lectures we define the set covering constraint (2b). This gives a mixed integer programming formulation of the problem described in Beyrouthy et al. (2007) for the single scenario case. Beyrouthy et al. prove that this model can easily be solved to optimality in linear time using the following greedy algorithm:

1. Sort lectures according to their size, $dem(c)$.
2. While lectures are unassigned to rooms:

   Take the largest unassigned lecture and assign it to a room with free timeslots. If no such room exists, add the smallest room that fits the lecture.

2.3. Teaching Periods problem

Another important factor that impacts on the capacity of a university is the number of timeslots that are available for teaching; this is explored in the Teaching Period problem, shown in Model 3, which decides how many timeslots there should be.

$$\begin{aligned}
\min \quad & f_{time} = \sum_{p\in\mathcal{P}} t_p && && (3a)\\
\text{s.t.} \quad & |\mathcal{R}_{\geq s}|\sum_{p\in\mathcal{P}} t_p \geq \sum_{c\in\mathcal{C}_{\geq s}} l(c) && \forall s \in \mathcal{S} && (3b)\\
& t_{p+1} - t_p \leq 0 && \forall p \in \mathcal{P} && (3c)\\
& t_p \in \mathbb{B} && \forall p \in \mathcal{P} && (3d)
\end{aligned}$$

Model 3. MIP-model for the Teaching Periods problem.

The mixed-integer model is seen in Model 3. A binary variable, t_p determines if timeslot $p \in \mathcal{P}$ is allowed. We then define the new objective f_{time} to be the total number of used timeslots as defined in (3a).

The set-covering constraint (3b) ensures there are enough timeslots, and that there is a timeslot and a room of matching size for each lecture.

Table 1
The indexing order of the timeslots.

Mon	Tue	Wed	Thu	Fri
1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
		...		

In practice, timeslots are chosen so that they are consecutive, as empty timeslots in the middle of the day are unwanted (except for other events like lunch breaks, meetings etc.). Constraint (3c) ensures that a new timeslot can only be used if the previous one is also used. Timeslots are ordered starting from Monday morning with the addition of a new timeslot each day, and then back to Monday; this order is illustrated in Table 1.

2.4. Room Planning vs. Quality

The previous Room Planning model does not consider if it is possible to create a feasible timetable as no timeslots are assigned. Neither does it take the quality of the resulting timetable into account. We will do this in the Room Planning vs. Quality model. Because the available rooms restrict the Quality model, it is necessary to take this into account when deciding what rooms to use, otherwise, it might result in a timetable of unacceptable poor quality. To analyze this, we use bi-objective optimization. Bi-objective optimization is a way to explore the trade-off between two objectives by generating multiple so-called Pareto-optimal solutions. A Pareto-optimal solution is a solution where one of the objectives cannot be improved without making the other worse; for further reading see Ehrgott (2000).

Model 4 is made by merging Models 1 and 2. The two models are connected by constraint (4c) which replaces (1b) and ensures that there can not be more lectures planned in a timeslot than rooms available. The r^+ variables have been added, because experiments show that they help the solver make stronger branches during the branch-and-bound search.

2.5. Teaching Periods vs. Quality

Similar to the previous problem, we want to explore the trade off between teaching periods and quality, as it is expected that more timeslots will give more freedom in the planning process. We do this by merging Models 1 and 3, which results in Model 5. The two models are connected by constraint (5c) which replaces (3b)

$$\begin{aligned}
\min \quad & f_{seats} : (2a) && && (4a)\\
& f_{qual} : (1a) && && (4b)\\
\text{s.t.} \quad & \sum_{c\in\mathcal{C}_{\geq s}} x_{c,p} - r_s^+ \leq 0 && \forall s \in \mathcal{S}, p \in \mathcal{P} && (4c)\\
& r_s^+ - \sum_{s'\in\mathcal{S}_{\geq s}} r_{s'} = 0 && \forall s \in \mathcal{S} && (4d)\\
& (1c) - (1m) && && (4e)\\
& (2b) - (2c) && && (4f)
\end{aligned}$$

Model 4. MIP-model for the Room Planning vs. Quality problem.

$$\begin{aligned}
\min \quad & f_{time} : (3a) && && (5a)\\
& f_{qual} : (1a) && && (5b)\\
\text{s.t.} \quad & x_{c,p} - t_p \leq 0 && \forall p \in \mathcal{P}, c \in \mathcal{C} && (5c)\\
& (1b) - (1m) && && (5d)\\
& (3c) - (3d) && && (5e)
\end{aligned}$$

Model 5. MIP-model for the Teaching Periods vs. Quality problem.

and states that lectures only can be assigned to a timeslot available for use.

2.6. Room Planning vs. Teaching Periods

When universities want to increase the number of courses above their current capacity, they can either build more rooms or increase the number of timeslots. In the Room Planning vs. Teaching Periods problem we investigate how these two objectives influence each other. This is done by merging Models 1, 4 and 5 into the new Model 6.

$$\begin{aligned}
\min \quad & f_{seats} : (2a) && (6a)\\
& f_{time} : (3a) && (6b)\\
\text{s.t.} \quad & (1c) - (1m) && (6c)\\
& (3c) - (3d) && (6d)\\
& (4c) - (4d) && (6e)\\
& (5c) && (6f)
\end{aligned}$$

Model 6. MIP-model for the Rooms vs. Teaching Periods problem.

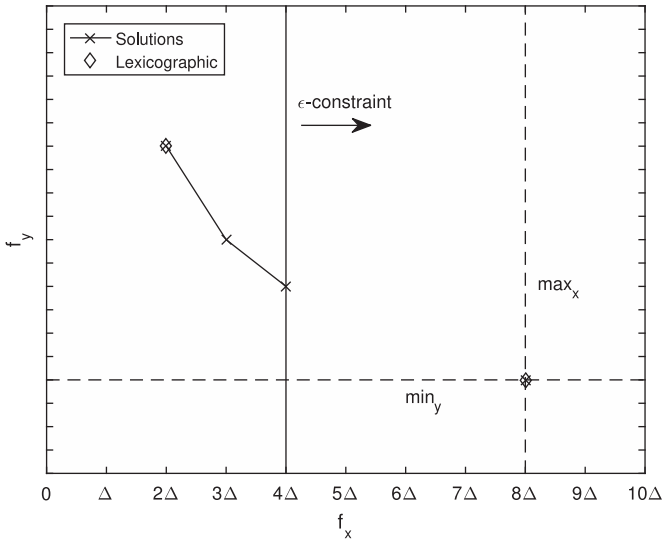

Fig. 2. An illustration of Algorithm 3.1, where the ϵ-constraint moves in steps of Δ. The algorithm terminates when it finds a solution where $\hat{f}_x > \max_x$ or $\hat{f}_y < \min_y$.

3. Solution methods

In this section, we propose an algorithm for solving our three bi-objective models. The original Quality problem in Model 1 is still hard to solve, and at this time 4 out of the 21 instances are still not solved to optimality; for the current status of this see http://satt.diegm.uniud.it/ctt/. The hardness also shows in Cacchiani, Caprara, Roberti, and Toth (2013) where six hours of running time are used for each instance to find good lower bounds. Because of this, we do not expect to solve these problems to optimality when making them bi-objective.

Many methods exist for solving bi-objective methods. A popular one is the ϵ-constraint method from Haimes, Ladson, and Wismer (1971) that has been applied to many different problems.

Both f_{seats} and f_{time} only takes discrete integer values and f_{seats} can only take values that depend on the size of potential rooms. From experiments, it is observed that the span between the minimum and maximum value of these two objectives is relatively small, which implies that a only a limited discrete set of values of f_{seats} and f_{time} exists. We, therefore, choose to use the ϵ-constraint method, where an artificial constraint is put on one of the objectives while minimizing the other. Because of the small set of potential values we only have to solve a small number of MIPs to explore all potential values of interest.

Let f_x be the objective with a limited discrete set of values, in our case f_{seats} or f_{time}, and let f_y be the other objective. The algorithm for solving objective f_x vs. f_y is shown on Algorithm 3.1 and illustrated in Fig. 2. The algorithm first calculates a lexicographic solution and uses this to put a feasibility limit on each objective. This is done by first finding a lower limit on f_y and then using this to calculate an upper feasibility limit for f_x. The main loop of the algorithm then starts by setting the ϵ-constraint right hand side to the minimum value of f_x and then repeatedly steps it up by the amount Δ. Limits on objectives are used to terminate the algorithm when one of them are exceeded. The reason for also having a limit on f_x and not only f_y is that we enforce a timelimit and do not expect to solve each problem to optimality. As shown in Lodi (2013), modern MIP solvers are heuristic and use randomness. Because of this, the algorithm can turn out in a lucky way and find a close to optimal value for $\min_y$, which it would then not be able to find later due to this randomness.

This method is generic for hard bi-objective problems where there is a limited set of values for one objective. Therefore, it can be used with different timetable constraints and quality measures. It should also be noted that if each subproblem is solved to optimality, this method will produce the optimal Pareto-front.

Algorithm 3.1 ϵ-constraint method - f_x vs. f_y.

1: **Parameters:** Δ ▷ Discretization of f_x
2: $min_y \leftarrow Minimize(f_y)$ ▷ Lower limit on f_y
3: Add constraint: $f_y = min_y$
4: $max_x \leftarrow Minimize(f_x)$ ▷ Upper limit on f_x
5: Remove constraint: $f_y = min_y$
6: $\epsilon \leftarrow Minimize(f_x)$
7: Add ϵ-constraint: $f_x \leq \epsilon$
8: **repeat**
9: $(\hat{f}_x, \hat{f}_y) \leftarrow Minimize(f_y)$ ▷ Find Pareto-solution
10: $\epsilon \leftarrow \epsilon + \Delta$
11: Update ϵ-constraint: $f_x \leq \epsilon$
12: **until** $\epsilon > max_x$ or $\hat{f}_y < min_y$ ▷ Only continue inside limits

4. Results

To show the results obtained by the proposed models and solution methods we use the instances from ITC2007 based on real-world examples from the University of Udine. All instances are available from tabu.diegm.uniud.it/ctt/. The complete source-code to make the computations is available on github.com/miclindahl/UniTimetabling. All computations are made on a 64 bit Windows machine with a 4 Gigahertz Intel Core i7 CPU and 32 Gigabytes of memory. To solve the integer programs, we use Gurobi 6.5 with standard settings.

Because this is a strategic problem not often solved, the timelimit for each iteration is set to 15 minutes. To generate the set $\mathcal{S}$ in the Room Planning problem, we use $\delta = 25$, that is a often practically used granularity.

As stated, each course is associated with a number of timeslots in which they cannot be taught. In the Teaching Periods problem we want to explore what happens when additional timeslots are added. To mark some of these timeslots as unavailable for courses we simulate it by using the following model: The probability that a course is unavailable in a new timeslot is equal to the fraction of timeslots that the course already is unavailable in. This way we

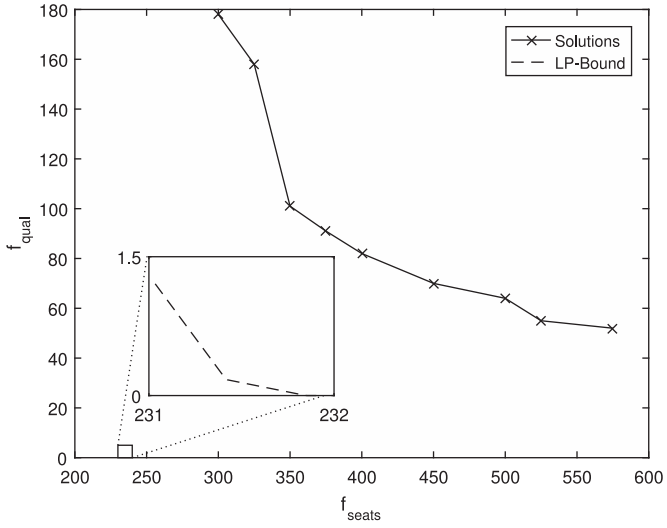

Fig. 3. Example from comp18 where it can be seen that the gap between the solution-frontier and the LP-relaxation is large and does therefore not give much help to the solver.

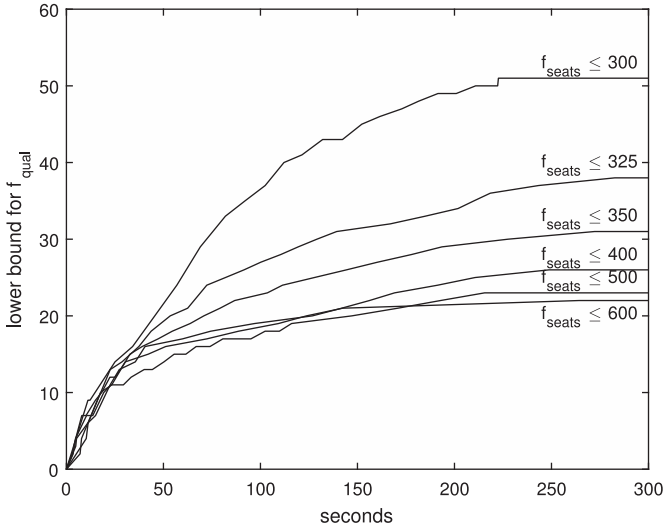

Fig. 4. Example from comp18 that shows the bounds improvement over time for different ϵ-constraints. It can be seen that it starts at zero, and it takes 30 seconds before the ϵ-constraint has a significant effect on the lower bound.

ensure that the new timeslots are realistic and have unavailabilities similar to the real data.

In this section, we will first discuss some attributes of the problem that makes it computationally difficult and afterward we show results for the three different problems. The trade-offs between the objectives will be reported with plots of the Pareto-solutions including bounds and a table with the lexicographic solutions.

4.1. Computational difficulties

Solving the mixed integer programs shows that the problems are difficult to solve. Two factors for this are poor LP solutions and the impact of the ϵ-constraint.

LP-Bound. MIP solvers start by relaxing the integer and binary constraints and solving the resulting LP relaxation that it uses to find integer solutions. Fig. 3 shows a plot of the Pareto-optimal solutions for the original integer version and the LP relaxation of Model 4. It is seen that there is a large gap between these two solutions. A weak LP relaxation makes it more difficult to find good integer solutions, as the relaxed solutions are used to guide the solver.

In Fig. 3 we observe that, as expected, we get better integer solutions (smaller f_{qual} values) as f_{seats} increases. We also observe that for all integer-feasible solutions the LP relaxation of f_{qual} is zero, which implies that the ϵ-constraint does not influence the LP-bound. This is seen in Fig. 4, where Gurobi's current lower bound on f_{qual} is plotted over time for different ϵ-constraints on f_{seats}. For every value of the ϵ-constraint, the lower bound starts at zero, and it is not until after thirty seconds that the bound differs significantly between the different values of the ϵ-constraint right hand side. This behavior helps explain the long run times we observed. The main contributor for this behaviour is the curriculum-compactness constraints, and it should be noted that if these are removed an optimal solution can be found in seconds.

Threshold behaviour. Another issue which creates difficulties is the so called *threshold behaviour*. An example of this is seen in Table 2, where comp18 is solved with three different ϵ-constraints on f_{seats}. The original problem with 405 seats is easy and can be solved to optimality in 33 seconds. We define the critical number of seats as the minimum number of seats that are needed for the problem to be feasible; this can easily be found by minimizing f_{seats} in Model 4. If the ϵ-constraint is reduced to the critical number, it becomes a lot more difficult and after 30 minutes of run time the gap is still 81%. If the number of available seats is then reduced by only one seat, the problem becomes infeasible and this can be proven in less than a second.

Table 2
Example of how the threshold effect shows on comp18. In the base case the optimal solution can be found in 33 seconds. When setting the maximum number of seats to the critical value, the problem gets difficult and the gap is still 81% after 30 minutes. If one seat is subtracted, the problem is proved infeasible instantly.

Case	ϵ-constraint		Time	Gap	Difficulty
Under-constrained	$f_{\text{seats}} \leq 405$		33 seconds	0%	Easy
Critical-constrained	$f_{\text{seats}} \leq 300$	$\Rightarrow$	30 minutes	81%	Hard
Over-constrained	$f_{\text{seats}} \leq 299$		0 second	–	Easy

This easy-hard-easy behaviour in timetabling is described in Beyrouthy et al. (2008) and is also seen in many other problem types. Beyrouthy et al. (2008) encounters this phase shift when the utilization has a critical value, and, as shown in Section 2.2, this is directly correlated with f_{seats}. If the number of seats is significantly bigger than the critical number of seats, this gives an under-constrained problem that is relatively easy to solve. Usually, heuristics can find good solutions to this problem because there is a lot of freedom to move lectures around without the problem becoming infeasible. In the other case, where the number of seats is smaller than the critical number, the problem is infeasible. This is also an easy problem, as a MIP-solver can prove this infeasibility fast, often in the pre-solve stage. The last case is the critical-constrained, which can be seen as *almost infeasible* problems. These are the difficult cases, as heuristics have difficulties in finding good solutions, and MIP-solvers will have trouble with large branch-and-bound trees.

4.2. Room Planning vs. Quality

To solve the bi-objective f_{seats} vs. f_{qual} optimization problem we use Model 4 and solve it by using Algorithm 3.1 where $f_x = f_{\text{seats}}$ is the objective with a limited set of values. To explore all combinations of the given rooms we set $\Delta = 25$. The Pareto fronts we generated for all 21 ITC datasets are shown in Fig. 5 together with the lower bounds returned by the solver; and the two lexicographic

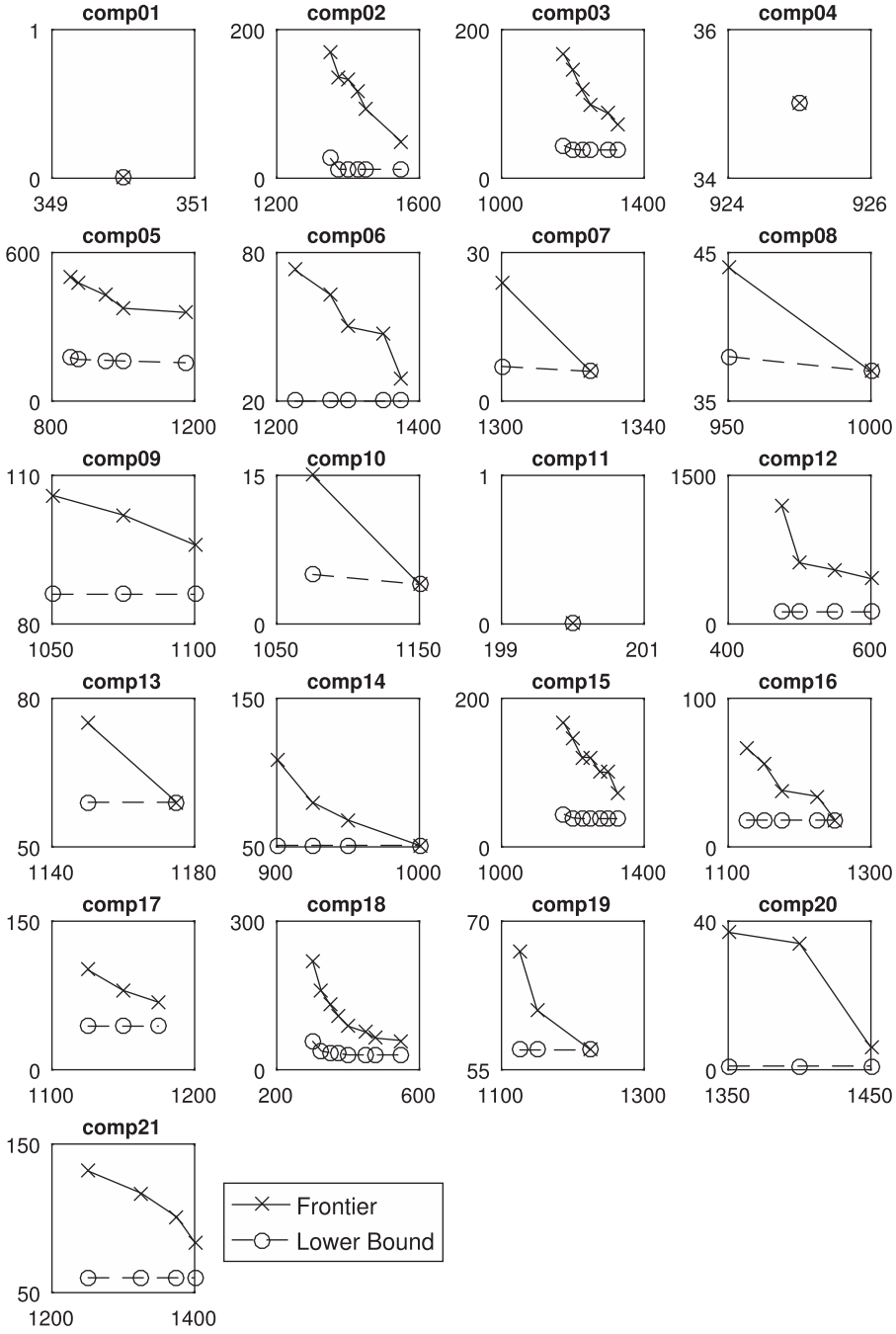

Fig. 5. The solution frontier for the Room Planning vs. Quality problem. The x-axis is f_{seats} and the y-axis is f_{qual}. In general, the gaps are large. Notice that three of the problems only have one solution.

solutions are shown in Table 3. The original number of seats and the total running time of generating the Pareto front in minutes is also given.

It is seen that comp01, comp04 and comp11 only have one solution meaning that there is no benefit from adding extra seats. The other instances only have little quality improvement from adding additional seats. But it can also be seen that the datasets comp02, comp03, comp15 and comp18 have a big trade-off between seats and quality. Comparing with the number of seats in the original instances it is seen that there is an average of 84% more capacity than where a feasible solution can be obtained.

The lower bounds from the solver are also shown. It can be seen that the gap is large due to the reasons discussed in Section 4.1.

Which rooms. To get more insights into why a certain solution is chosen as the optimal room configuration we analyze the solutions. A metric for the overcapacity can be deduced from Constraint (2b), allowing us to calculate the $overcapacity_s$ for a given size $s \in \mathcal{S}$ as follows:

$$\text{overcapacity}_s = \frac{|\mathcal{P}| \sum_{s \in \mathcal{S}_{\geq s}} r_s}{\sum_{c \in \mathcal{C}_{\geq s}} l(c)} - 1$$

This metric gives the average number of empty timeslots with available rooms for each lecture. A negative number will violate Constraint (2b) and, will therefore, be infeasible. A positive number indicates how much freedom there is to move lectures around. An example of the overcapacity for the solutions is seen in Table 4. It is seen that the solver finds solutions that add additional room

Table 3
The two lexicographic solutions for the Room Planning vs. Quality problem and the original number of seats.

Instance		Runtime	Fewest seats		Best quality	
	f_{seats}	Min.	f_{seat}	f_{qual}	f_{seat}	f_{qual}
comp01	389	0	350	0	350	0
comp02	2350	165	1350	170	1550	49
comp03	2360	120	1175	167	1325	72
comp04	2119	7	925	35	925	35
comp05	1083	226	850	502	1175	358
comp06	2204	120	1225	73	1375	29
comp07	2342	31	1300	24	1325	6
comp08	1936	45	950	44	1000	37
comp09	2440	60	1050	106	1100	96
comp10	2179	75	1075	15	1150	4
comp11	222	0	200	0	200	0
comp12	717	105	475	1193	600	458
comp13	2282	33	1150	75	1175	59
comp14	1913	83	900	109	1000	51
comp15	2360	120	1175	167	1325	72
comp16	2366	96	1125	67	1250	18
comp17	2199	60	1125	101	1175	68
comp18	543	180	300	221	550	59
comp19	2350	91	1125	67	1225	57
comp20	2222	90	1350	37	1450	6
comp21	2480	190	1250	132	1400	84

Table 4
The *overcapacity* for different solutions to `Comp18`. It can be seen that average overcapacity (avg.) increases by increasing f_{seats}. $\min_1$ and $\min_2$ denote the smallest and second smallest value. It is seen that, in general, when more seats are added the minimum overcapacity is also increased.

	Overcapacity$_s$							Summary		
f_{seats}	$s=$	0	25	50	75	100	125	Avg.	$\min_1$	$\min_2$
300	0.30		0.33	0.24	1.57	11.00	11.00	4.07	0.24	0.30
325	0.30		0.33	1.48	1.57	11.00	11.00	4.28	0.30	0.33
350	0.83		0.33	0.24	1.57	11.00	11.00	4.16	0.24	0.33
375	0.57		0.78	1.48	1.57	11.00	11.00	4.40	0.57	0.78
400	0.57		1.22	1.48	1.57	11.00	11.00	4.47	0.57	1.22
450	0.83		1.67	1.48	1.57	11.00	11.00	4.59	0.83	1.48
475	0.83		1.67	2.72	1.57	11.00	11.00	4.80	0.83	1.57
500	1.09		2.11	1.48	1.57	11.00	11.00	4.71	1.09	1.48
525	1.09		2.11	2.72	1.57	11.00	11.00	4.92	1.09	1.57
550	1.35		2.11	2.72	1.57	11.00	11.00	4.96	1.35	1.57
575	1.35		2.11	2.72	4.14	11.00	11.00	5.39	1.35	2.11

capacity where the overcapacity is smallest. This metric can be used as a rule of thumb for managers to give them an indication about what rooms to add to increase quality.

4.3. Teaching Periods vs. Quality

To analyze the trade-off between f_{time} and f_{qual} we use Model 5 and solve it with Algorithm 3.1. We apply the ϵ-constraint on f_{time} and use $\Delta = 1$ so that we add only one timeslot in each iteration to investigate all possibilities. Pareto plots showing the trade-off between f_{time} and f_{qual} for all 21 ITC instances are shown in Fig. 6. The two lexicographic solutions are given in Table 5 together with the running time. Similar to the previous results, it can be seen that six of the datasets do not benefit from getting extra timeslots. On the other datasets there are significant quality improvements generated by adding extra timeslots. Comparing with the original instances, it is seen that they have an average of 13% more timeslots than the minimum.

The lower bounds are closer to the solutions and seven problems are solved to optimality. Some of the instances do, however, still have large gaps.

Table 5
The lexicographic solutions for the Teaching Periods vs. Quality problem. The run times (in minutes) and the original number of timeslots (f_{time}) are also given for each instance.

Instance		Runtime	Fewest timeslots		Best quality	
	f_{time}	min.	f_{time}	f_{qual}	f_{time}	f_{qual}
comp01	30	0	32	8	36	0
comp02	25	255	22	49	36	30
comp03	25	45	23	80	23	80
comp04	25	22	20	48	44	32
comp05	36	75	33	369	35	344
comp06	25	315	21	48	39	25
comp07	25	35	22	6	22	6
comp08	25	21	21	45	45	32
comp09	25	58	23	100	24	96
comp10	25	28	21	8	25	4
comp11	45	0	40	0	40	0
comp12	36	180	27	553	36	385
comp13	25	128	19	70	45	48
comp14	25	343	20	53	44	43
comp15	25	45	23	80	23	80
comp16	25	95	19	48	24	18
comp17	25	45	23	65	23	65
comp18	36	195	17	192	27	72
comp19	25	150	23	58	45	42
comp20	25	90	24	12	27	8
comp21	25	45	24	97	24	97

Table 6
The lexicographic solutions of the results of the Teaching Periods vs. Room Planning problem.

Instance	Runtime	Fewest timeslots		Fewest seats	
	min.	f_{time}	f_{seat}	f_{time}	f_{seat}
comp01	0	24	400	54	250
comp02	0	22	1550	49	775
comp03	0	23	1250	50	700
comp04	0	20	1125	48	525
comp05	0	33	850	54	600
comp06	0	18	1625	48	675
comp07	0	20	1575	50	675
comp08	0	21	1125	48	550
comp09	0	23	1100	47	575
comp10	0	19	1375	47	625
comp11	0	28	275	55	150
comp12	0	27	575	59	300
comp13	0	18	1525	50	650
comp14	0	20	1100	46	525
comp15	0	23	1250	50	700
comp16	0	19	1475	46	625
comp17	0	23	1200	49	625
comp18	0	17	475	53	225
comp19	0	23	1175	47	625
comp20	0	18	1800	49	750
comp21	0	24	1250	48	700

4.4. Teaching Periods vs. Room Planning

To analyze how the two objectives, f_{time} and f_{seats} affect each other we show the trade-off by using the Model 6.

Fig. 7 shows the Pareto front and the two lexicographic solutions, and the running time are shown in Table 6. It is seen that this problem is faster to solver, and that optimal Pareto frontiers can be generated in less than a minute for all instances. This also shows that the soft constraints have the biggest impact on the difficulty of solving these problems.

The trade-off between f_{time} and f_{seats} is significant on all datasets. Because the utilization U is linear proportional with $|\mathcal{P}| \sum_{r \in \mathcal{R}} \text{cap}(r)$ the trade-off is almost linear.

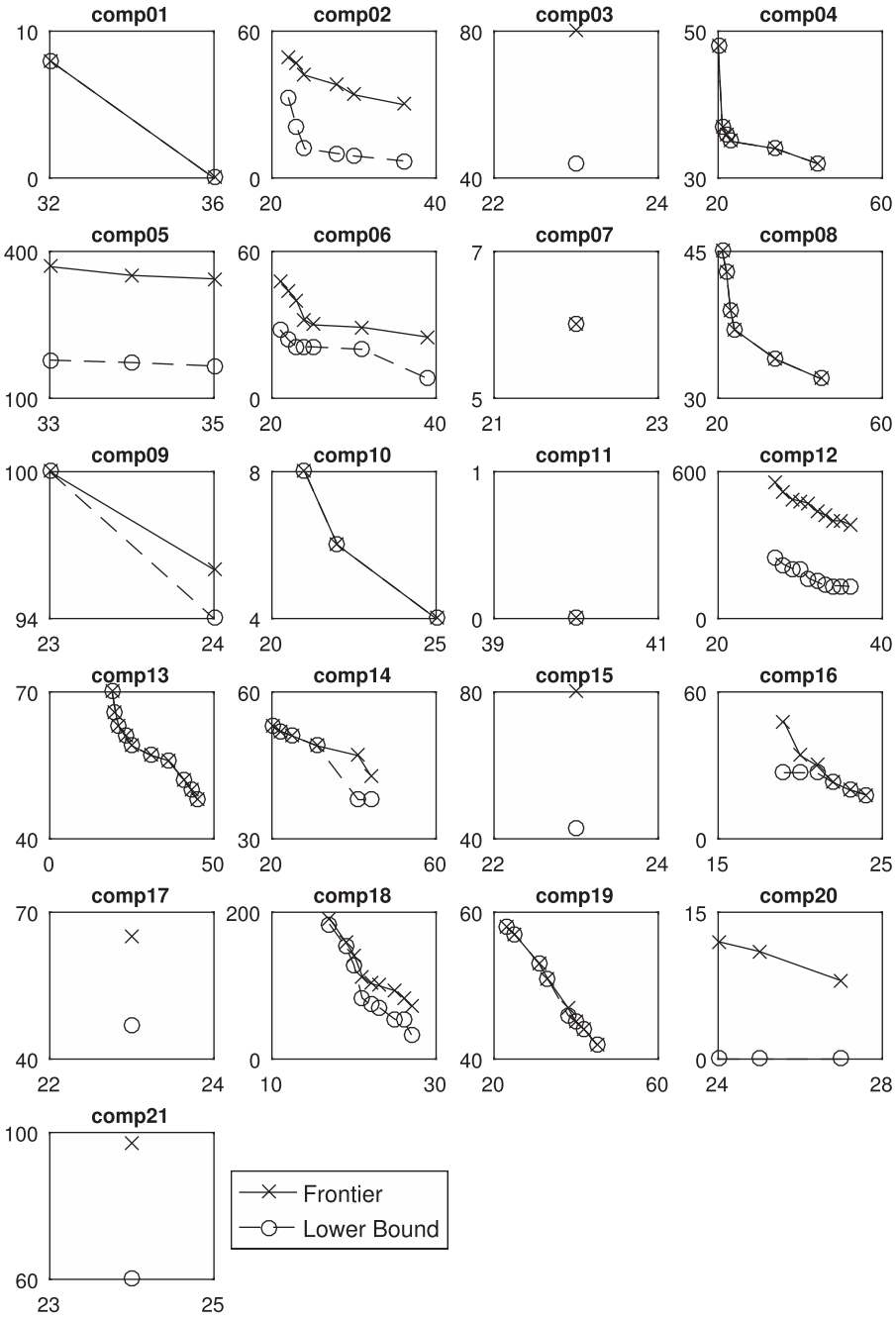

Fig. 6. The solution frontiers for the Teaching Periods vs. Quality problem. The x-axis is f_{time} and the y-axis is f_{qual}. The gaps are smaller than in the Room Planning vs. Quality problem. Six of the instances only have one Pareto-optimal solution.

5. Conclusions

University management needs to continuously manage their resources efficiently. Resources like teachers, rooms and teaching periods cannot be increased or decreased on short notice. These management decisions have a high impact on the cost of running a university and also on how good the resulting timetables will be.

Analyzing how strategic decisions affect timetabling quality is both important to help make these high-impact decisions better and to get a better understanding of the structure of the timetabling problem and see how the availability of these resources affects the solution.

This paper presents the first attempt to investigate these trade-offs. We propose a method that can potentially solve these problems to optimality and create the entire frontier. Our method is applicable to most timetabling problems.

Finally, we have shown how the three objectives – rooms, teaching periods, and quality – have an impact on one another, and we have shown that this interaction between competing objectives varies between the different instances. We have analyzed optimal room profiles to give managers a rule of thumb that can be used to help make decisions better.

Future research. This paper explored two soft constraints as quality metrics that both focused on the timeslots in which lectures were planned. Exploring these trade-offs with other metrics could be interesting. Another simplification is that all rooms are considered to be identical except for capacity. In practice, rooms often have

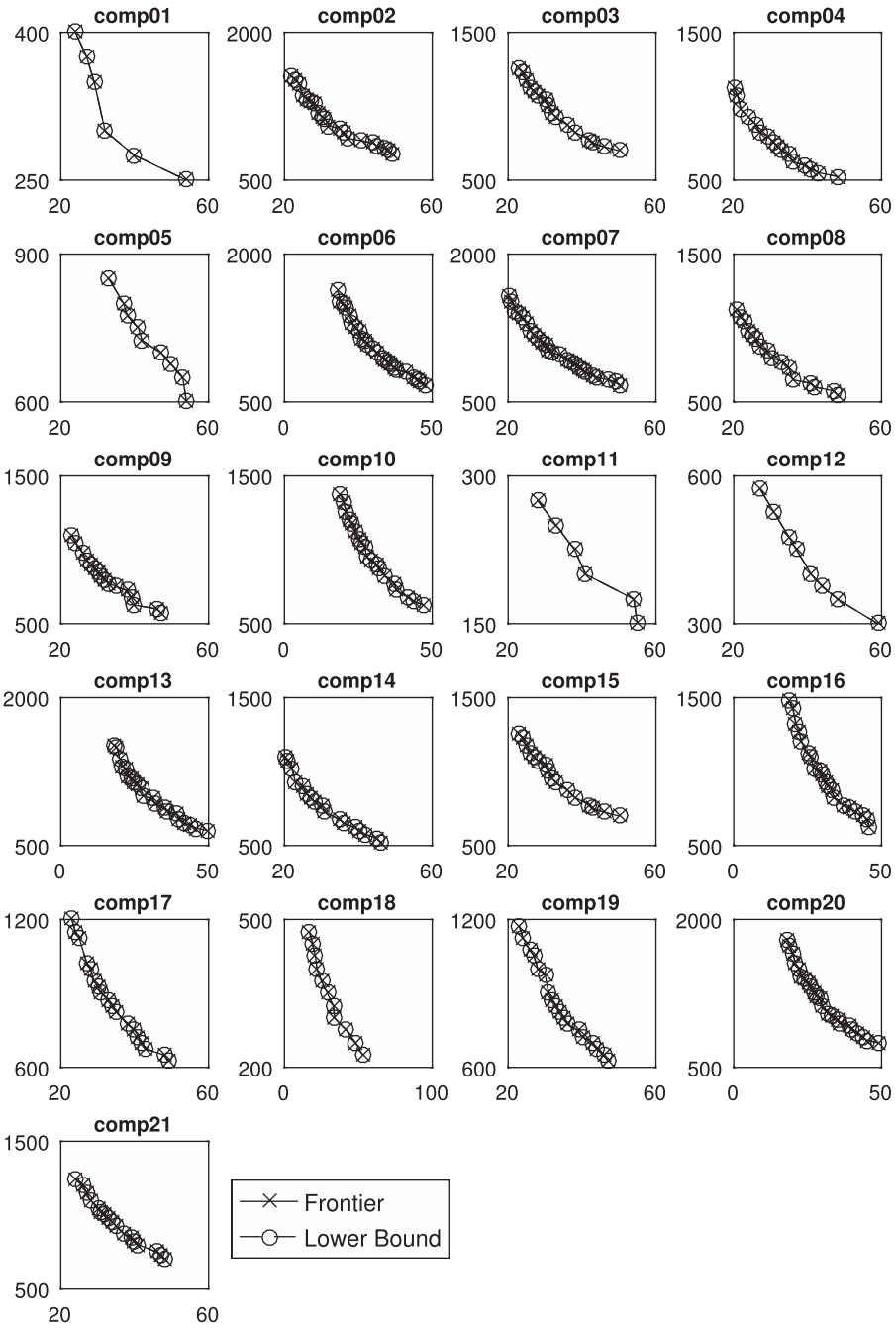

Fig. 7. The solution frontiers for the Teaching Periods vs. Room Planning problem. The x-axis is f_{time} and the y-axis is f_{seats}. Notice that all of these solutions are optimal. There is an almost linear relationship between these two objectives.

certain attributes requested by some courses. Many campuses are so large that the location of the rooms matters; this would give some different trade-offs that would be of interest.

The authors hope that this paper can generate interest in finding and answering more of the strategic questions that are occurring in timetabling.

Acknowledgment

The authors would like to thank the organizers of ITC-2007 for providing a formal problem description of CB-CTT as well as benchmark instances. The authors would also like to thank Alex Bonutti, Luca Di Gaspero and Andrea Schaerf for creating and maintaining the website for instances and solutions to CB-CTT.

References

Bettinelli, A., Cacchiani, V., Roberti, R., & Toth, P. (2015). An overview of curriculum-based course timetabling. *TOP, 23*, 1–37 https://link.springer.com/article/10.1007/s11750-015-0366-z.

Beyrouthy, C., Burke, E. K., Landa-Silva, D., McCollum, B., McMullan, P., & Parkes, A. J. (2007). Improving the room-size profiles of university teaching space. In *Proceedings of the Dagstuhl seminar on "cutting, packing, layout and space allocation"*. Citeseer.

Beyrouthy, C., Burke, E. K., Landa-Silva, D., McCollum, B., McMullan, P., & Parkes, A. J. (2008). Threshold effects in the teaching space allocation problem with splitting. *Technical Report*. University of Nottingham.

Beyrouthy, C., Burke, E. K., Landa-Silva, D., McCollum, B., McMullan, P., & Parkes, A. J. (2009). Towards improving the utilization of university teaching space. *Journal of the Operational Research Society, 60*(1), 130–143.

Beyrouthy, C., Burke, E. K., McCollum, B., McMullan, P., & Parkes, A. J. (2010). University space planning and space-type profiles. *Journal of Scheduling, 13*(4), 363–374.

Cacchiani, V., Caprara, A., Roberti, R., & Toth, P. (2013). A new lower bound for curriculum-based course timetabling. *Computers & Operations Research, 40*(10), 2466–2477.

Di Gaspero, L., McCollum, B., & Schaerf, A. (2007). The second international timetabling competition (ITC-2007): Curriculum-based course timetabling (track 3). *Technical Report*. School of Electronics, Electrical Engineering and Computer Science, Queens University SARC Building, Belfast, United Kingdom.

Ehrgott, M. (2000). *Multicriteria optimization*. Springer.

Fizzano, P., & Swanson, S. (2000). Scheduling classes on a college campus. *Computational Optimization and Applications, 16*(3), 279–294.

Haimes, Y. Y., Ladson, L., & Wismer, D. A. (1971). *Bicriterion formulation of problems of integrated system identification and system optimization, SMC-1*(3), 296–297 https://ieeexplore.ieee.org/document/4308298.

Kingston, J. H. (2013). Educational timetabling. In A. S. Uyar, E. Ozcan, & N. Urquhart (Eds.), *Automated scheduling and planning.* In *Studies in Computational Intelligence: 505* (pp. 91–108). Springer Berlin Heidelberg.

Kristiansen, S., & Stidsen, T. R. (2013). A comprehensive study of educational timetabling - A survey. *Technical Report 8, 2013*. DTU Management Engineering, Technical University of Denmark.

Lach, G., & Lübbecke, M. (2012). Curriculum based course timetabling: new solutions to udine benchmark instances. *Annals of Operations Research, 194*, 255–272.

Lodi, A. (2013). The heuristic (dark) side of mip solvers. In E.-G. Talbi (Ed.), *Hybrid metaheuristics*. In *Studies In Computational Intelligence: vol. 434* (pp. 273–284). Springer Berlin Heidelberg.

Muller, T., Rudová, H., & Barták, R. (2005). Minimal perturbation problem in course timetabling. In E. Burke, & M. Trick (Eds.), *Practice and theory of automated timetabling v*. In *Lecture Notes in Computer Science: vol. 3616* (pp. 126–146). Springer Berlin Heidelberg.

Phillips, A. E., Walker, C. G., Ehrgott, M., & Ryan, D. M. (2014). Integer programming for minimal perturbation problems in university course timetabling. In *Proceedings of the 10th international conference of the practice and theory of automated timetabling patat 2014, 26–29 august 2014, York, United Kingdom*.

Wren, A. (1996). Scheduling, timetabling and rostering a special relationship?. In E. Burke, & P. Ross (Eds.), *Practice and theory of automated timetabling* In *Lecture Notes in Computer Science: 1153* (pp. 46–75). Springer Berlin / Heidelberg.