Computers & Operations Research 43 (2014) 36–49

Contents lists available at ScienceDirect

Computers & Operations Research

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

A Two-Stage Decomposition of High School Timetabling applied to cases in Denmark

Matias Sørensen [a,b,*], Florian H.W. Dahms [c]

[a] Section of Operations Research, Department of Management Engineering, Technical University of Denmark, DK-2800 Kgs. Lyngby, Denmark
[b] MaCom A/S, Vesterbrogade 48, 1., DK-1620 Copenhagen V, Denmark
[c] Chair of Operations Research, RWTH Aachen University, Kackertstrasse 7, 52072 Aachen, Germany

ARTICLE INFO

Available online 5 September 2013

Keywords:
High school timetabling
Integer programming
Decomposition
Bipartite matching

ABSTRACT

Integer Programming (IP) has been used to model educational timetabling problems since the very early days of Operations Research. It is well recognized that these IP models in general are hard to solve, and this area of research is dominated by heuristic solution approaches. In this paper a *Two-Stage Decomposition* of an IP model for a practical case of high school timetabling is shown. This particular timetabling problem consists of assigning lectures to both a timeslot and a classroom, which is modeled using a very large amount of binary variables. The decomposition splits this model into two separate problems (Stage I and Stage II) with far less variables. These two separate problems are solved in sequence, such that the solution for the Stage I model is given as input to the Stage II model, implying that irreversible decisions are made in Stage I. However, the objective of the Stage II model is partly incorporated in the Stage I model by exploiting that Stage II can be seen as a *minimum weight maximum matching* problem in a bipartite graph. This theoretically strengthens the decomposition in terms of global optimality. The approach relies on Hall's theorem for the existence of matchings in bipartite graphs, which in its basic form yields an exponential amount of constraints in the Stage I model. However, it is shown that only a small subset of these constraints is needed, making the decomposition tractable in practice for IP solvers. To evaluate the decomposition, 100 real-life problem instances from the database of the high school ERP system Lectio are used. Computational results show that the decomposition performs significantly better than solving the original IP, in terms of both found solutions and bounds.

1. Introduction

Integer Programming (IP) has been used to model educational timetabling problems since the very early days of Operations Research (see e.g. [16,19]). It is well recognized that these IP models in general are hard to solve (most forms of educational timetabling are in fact $\mathcal{NP}$−hard [4]), and this area of research is dominated by heuristic solution approaches.

In this paper a large IP model for a real-world case of high school timetabling is considered, which has previously been shown to be a challenge for state-of-the-art MIP solvers. We consider a basic version of this IP, which includes the essential constraints of most timetabling problems. An innovative decomposition of this model is shown, which proves to be more efficient to solve.

When facing a hard IP model, decomposition is a commonly used tool to help speed up the solution procedure. Perhaps the most successful decomposition method in recent years is Column Generation (CG). However, not many papers on CG and timetabling models are found in the literature, and it seems that only relatively small instances have been attempted. Papoutsis et al. [23] use CG to solve a Greek case of high school timetabling, with the largest instance containing 9 class section, 21 teachers and 306 teaching hours. Santos et al. [28] handle larger instances, but only generate lower bounds. Qualizza and Serafini [27] describe a CG procedure for a university timetabling problem with 63 courses and 25 timeslots. The real-world instances considered in this paper are of much larger size.

A crucial part of a CG procedure is the identification of a block-diagonal structure in the problem, otherwise the CG procedure is most likely not efficient. For the high school timetabling problem described in this paper, it has not been possible to identify such a structure. Therefore this paper shows a different type of decomposition, a *Two-Stage Decomposition* (TSD). Such an approach was first used for timetabling applications in Lach and Lübbecke [17,18] with great success for the *curriculum-based university course* timetabling problem. The goal of this paper will be to modify the aforementioned approach to be applicable for the high school timetabling problem – giving special attention to the high school system in Denmark.

* Corresponding author at: Section of Operations Research, Department of Management Engineering, Technical University of Denmark, DK-2800 Kgs. Lyngby, Denmark. Tel.: +45 45254800.
E-mail addresses: msso@dtu.dk, sorensen.matias@gmail.com (M. Sørensen).

0305-0548/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cor.2013.08.025

The considered timetabling problem essentially consists of assigning lectures to rooms and timeslots, which is commonly modeled using a very large amount of binary variables. There are three key points to the TSD:

- By substitution, the total amount of variables is significantly reduced, while linearity is maintained.
- Instead of solving the entire model at once, it can be solved in a two-stage fashion. i.e. both the set of variables and constraints are divided into two distinct sets, corresponding to two smaller IPs (denoted *Stage I* and *Stage II*, respectively).
- It will be evident that, except for two soft-constraints, this decomposition maintains optimality of the original model.

The outline of the TSD is to first solve Stage I, which provides a solution where lectures are assigned to timeslots. This partial solution is given as input to Stage II, which will assign rooms to the lectures, obtaining a solution for the original problem. The drawback of this decomposition is that the timeslots assigned to lectures in Stage I are considered as fixed by the Stage II model, which might prevent an optimal allocation of rooms to lectures. However, by exploiting the structure of the Stage II model, the Stage I model can be constrained in such a way that some penalties for assigning rooms to lectures are handled implicitly. Note that if all penalties for room assigning could be handled implicitly, the approach would be exact. However, two soft-constraints are not fully incorporated, so only a lower bound on the room penalties is known by the Stage I model. In fact, one of these soft-constraints are not handled at all by the described approach. Despite this, it seems likely that incorporating this lower bound in the Stage I model will provide better results overall (assuming that computing the lower bound does not have very bad influence on the computational efforts of the used IP solver). So instead of the Stage I model being completely unaware of the penalties for room allocation, it seems better to at least incorporate some of them. Furthermore, the decomposition of the problem into two smaller problems presents a big advantage in terms of reduction in the number of variables. Therefore the overall benefits of the TSD out-weight the downsides, and computational results will show that it is indeed way more effective than solving the original IP.

The contributions of this paper are the following: (1) It is shown that the approach from Lach and Lübbecke [17] can also be applied to a high school timetabling problem originating from a practical setting, and by extensive computational results it is argued that the TSD is more effective than solving the original IP. Notice that a similar decomposition is briefly mentioned in Sørensen and Stidsen [31] for the same high school timetabling problem, but this paper enhances the approach such that the theoretical maximum gap from optimality is narrowed. The presented approach turns out to be the most efficient exact algorithm for the problem so far. (2) Generally, it is shown how this type of decomposition can be applied to models with set-packing structure, by modifying the underlying equations originating from Hall's Theorem for matchings in bipartite graphs. (3) It is shown how the room-priorities of lectures can be handled, by adding a lower bound on the corresponding penalties to the Stage I model. This facilitates the quality of the solutions found, as shown by the computational results.

We expect that the basic structure required for applying the TSD can be found in other timetabling problems as well, and therefore the decomposition can potentially be used more broadly than the case of high school timetabling shown in this paper. This seems likely because the essential constraints used in the decomposition are among the most common ones found in timetabling problems.

The paper is structured as follows. First related papers are described in Section 2. The basic IP model is introduced in Section 3, including the essential constraints. Section 4 shows the TSD of this model, and derives the lower bound on room allocation penalties for the Stage I model. Section 5 extends the model so it encapsulates a practical version of the high school timetabling problem, defined by the online high school administration system Lectio. Section 6 shows computational results, comparing the decomposition to previous approaches for 100 problem instances taken from the Lectio database. Section 7 concludes on our findings.

2. Related work

Integer Programming has been used to model various educational timetabling problems. However, heuristics are still the most popular method for these problems, see surveys [29,24]. In terms of IP, de Werra [12] describes what is called 'a simple model' for the class-teacher problem, and existence of solutions is proven under certain circumstances using graph theoretical models. The problems considered are feasibility problems, and soft constraints are not added to the models. Birbas et al. [5] describes a 'fully defined' IP model for Greek secondary schools, which is evaluated on five different schools with success. Avella et al. [1] formulates an IP model which is used to solve small instances of various origin. The IP is solved within a VLSN algorithm, with good results.

For the related university course timetabling problem, Daskalaki et al. [11] presents a model which schedules courses to timeslots and classrooms, using many so called *operational rules*. Three different problem instances of significant size are all solved to optimality using CPLEX. MirHassani [21] describes the problem for an Iranian university, and reports good results by applying the XA solver. In Dimopoulou and Miliotis [13] an IP model is used to solve the timetabling problem for The Athens University of Economics and Business.

Decomposition of IP models for educational timetabling is not a very well researched topic. Burke et al. [7] state that: *In the timetabling community, the "times first, rooms second" decomposition is a standard procedure*. However, it seems that this procedure has not been applied much in context of IP models. Burke and Newall [8] apply the procedure in context of an Evolutionary Algorithm for Examination Timetabling. In terms of multistage-decompositions, the importance of Lach and Lübbecke [17,18] has already been discussed. Carter [9] presents an interesting decomposition algorithm for course timetabling with elective courses. Stating the problem in terms of a vertex coloring problem facilitates the decomposition of the graph by cliques, such that the subproblem defined by each clique is solved separately.

In Burke et al. [7], experiments are conducted on disabling different combinations of soft-constraint penalties of the *Udine Course Timetabling Problem*, including one where all room penalties on room allocation are disabled. Thereby a similar decomposition to that of Lach and Lübbecke [18] is obtained.

Daskalaki and Birbas [10] presents an approach for university timetabling, where courses are first assigned to days (skipping some requirements for compactness), and in the following stage the timetable for each day is treated locally (enforcing the compactness). Convincing computational results are shown. In Birbas et al. [6], a high school timetabling problem is solved by first allocating 'work shifts' to teachers, and then solving the actual timetabling problem. This is related to the type of decomposition performed in this paper. Badri [2] uses a related approach for university course timetabling, where *faculties* are first assigned to courses, and then faculties are assigned to timeslots. However the problems solved are tiny.

Recently, high school timetabling received attention in the *International Timetabling Competition 2011* (ITC2011), see Post et al. [25]. This competition built upon an uniform format for formulating problem instances (and their solutions), known as *XHSTT* [26]. Currently, around 50 problem instances are available in this format. The problem considered in this paper deviates from the XHSTT format in several important ways, which is beyond the scope of this section to elaborate on. Even though many researchers participated in ITC2011, it seems that all were applying heuristics.

3. An integer programming model for high school timetabling

As the origin for our approach lies the IP model presented in Sørensen and Stidsen [31]. To make a clear presentation of the TSD, this IP model is reduced to its essential parts, which is described in the following. In Section 5, the full IP model is shown in context of the TSD.

A set of *events* $\mathcal{E}$ is given. Each event generally represents one lecture, which is defined as a meeting between specific *resources*, with a certain subject as teaching-objective. The set of resources is denoted $\mathcal{A}$. The goal of the high school timetabling problem is to assign each event to a room and to a timeslot, such that no conflicts among resources occur. The set of rooms and timeslots are denoted $\mathcal{R}$ and $\mathcal{T}$, respectively. The decision variable $x_{e,r,t} \in \{0,1\}$ takes value 1 if event $e \in \mathcal{E}$ is assigned room $r \in \mathcal{R}$ and timeslot $t \in \mathcal{T}$. To ensure a feasible solution exists, both the set of timeslots $\mathcal{T}$ and the set of rooms $\mathcal{R}$ are extended with a single dummy element, i.e. $\mathcal{T} = \{\mathcal{T} \cup t_D\}$ and $\mathcal{R} = \{\mathcal{R} \cup r_D\}$. This should be interpreted in the way that assigning to these dummy-elements actually means that no timeslot/room was assigned to the event. Thereby the goal of the IP is to assign as many events *as possible* to a timeslot and/or a room. From a practical point of view this is desirable, as the model is used in a decision support context where it might not be evident how to handle infeasibility. $\phi_{e,t} \in \mathbb{R}^+$ denotes the penalty for assigning event $e \in \mathcal{E}$ to timeslot $t \in \mathcal{T}$, and $\pi_{e,r} \in \mathbb{R}^+$ denotes the penalty for assigning event $e \in \mathcal{E}$ to room $r \in \mathcal{R}$. Major penalties are given for assignments to the dummy-elements, i.e.

$$\phi_{e,t_D} \gg \phi_{e,t} \quad \forall e \in \mathcal{E}, t \in \mathcal{T} \backslash t_D \tag{1}$$

$$\pi_{e,r_D} \gg \pi_{e,r} \quad \forall e \in \mathcal{E}, r \in \mathcal{R} \backslash r_D \tag{2}$$

A room might be unavailable in certain timeslots, indicated by the binary parameter $G_{r,t} \in \{0,1\}$, which takes value 1 if room $r \in \mathcal{R}$ is available in timeslot $t \in \mathcal{T}$, and 0 otherwise. Furthermore, a set of eligible rooms exists for each event. Let parameter $K_{e,r} \in \{0,1\}$ take value 1 if event $e \in \mathcal{E}$ can take place in room $r \in \mathcal{R}$, and 0 otherwise. Each event requires a fixed set of resources. Let $E_a^{'}, a \in \mathcal{A}$, denote the set of events where resource a participates.

We include in the model a set of constraints which will be described later, denoted by the constraint-set P_{other}, slightly abusing notation. These constraints define various other important criteria, such as forbidden timeslots for certain events, events which must be placed in the same timeslots, etc. Since these constraints are not required for describing the decomposition, their definitions are postponed to Section 5. We allow the set of constraints P_{other} to also denote soft-constraints (i.e. constraints which result in a weighted penalty in the objective function if it is not fulfilled). Thereby these constraints contain all necessary conditions for modeling the timetabling instances in question, and represent a large set of distinct types of constraints. It will be argued in the next section that these constraints can be handled in the decomposition such that optimality of the IP model is not lost, with one exception.

Model (3) shows the IP model.

IP Model for High School Timetabling (3)

$$\min \quad w = \sum_{e \in \mathcal{E}, r \in \mathcal{R}, t \in \mathcal{T}} (\phi_{e,t} + \pi_{e,r}) x_{e,r,t} \tag{3a}$$

s.t.

$$\text{(one time/room)} \quad \sum_{r \in \mathcal{R}, t \in \mathcal{T}} x_{e,r,t} = 1 \quad \forall e \in \mathcal{E} \tag{3b}$$

$$\text{(resource conf.)} \quad \sum_{r \in \mathcal{R}, e \in E_a^{'}} x_{e,r,t} \leq 1 \quad \forall a \in \mathcal{A},\ t \in \mathcal{T} \backslash t_D \tag{3c}$$

$$\text{(room conf.)} \quad \sum_{e \in \mathcal{E}} x_{e,r,t} \leq G_{r,t} \ \forall r \in \mathcal{R} \backslash r_D, t \in \mathcal{T} \backslash t_D \tag{3d}$$

$$\text{(eligible rooms.)} \quad \sum_{t \in \mathcal{T}} x_{e,r,t} \leq K_{e,r} \ \forall e \in \mathcal{E},\ r \in \mathcal{R} \tag{3e}$$

$$x_{e,r,t} \in P_{\text{other}} \tag{3f}$$

$$x_{e,r,t} \in \{0,1\} \tag{3g}$$

The objective of the model is to minimize the overall penalty for assignments, given by (3a). Constraint (3b) specifies that each event must be assigned exactly one timeslot and one room. Events which require the same resource cannot be scheduled simultaneously (except in the dummy-timeslot), which is ensured by constraint (3c). A room cannot be used by more than one event in each timeslot. This is specified in constraint (3d). The requirement for eligible rooms is specified in constraint (3e). Constraint (3f) specifies that constraints P_{other} should be respected.

Theorem 3.1. *The High School Timetabling Problem as specified in (3) is $\mathcal{NP}-$hard.*

Proof. We conduct a reduction from Vertex Coloring. Let $G = (V, E)$ be an arbitrary graph and k be an arbitrary number. The question of the coloring problem would now be whether it is possible to color G with k colors, such that no two adjacent vertices share the same color.

Now construct a High School Timetabling instance in the following way:

- Let there be an event for every vertex, i.e. $\mathcal{E} = V$.
- Make sure there are enough rooms for all events, therefore create a room for every event (i.e. $|\mathcal{R}| = |\mathcal{E}|$) and make sure all events fit in all rooms (i.e. $K_{e,r} = 1 \forall e \in \mathcal{E}, r \in \mathcal{R}$), and that all rooms are available in all timeslots (i.e. $G_{r,t} = 1 \forall e \in \mathcal{E}, r \in \mathcal{R}$).
- For every edge $\{v_1, v_2\} \in E$ we create a resource in $\mathcal{A}$ (i.e. $\mathcal{A} = E$). The events using this resource will be the vertices connected by the edge (i.e. $E_{\{v_1,v_2\}}^{'} = \{v_1, v_2\}$).
- We use exactly k timeslots (i.e. $\mathcal{T} = \{1, \ldots, k\}$).
- The additional constraints P_{other} can be dropped without loss of generality, as we impose no other restrictions on the timetabling instance. Also as we only search for a feasible solution, the soft constraints can easily be ignored.

Now we have a direct relation between the Vertex Coloring problem and the new timetabling instance. A solution of one problem can be transformed into a solution of the other by translating the colors of the vertices into timeslots, and vice versa. The rooms pose no restriction as every event can be scheduled in its own room.

Therefore solving the timetabling instance would result in solving the Vertex Coloring Problem, and the High School Timetabling problem is $\mathcal{NP}-$hard.

Here we remark that Model (3) encapsulates many of the basic constraints required by most timetabling problems. If we for

instance consider the XHSTT format, the basic requirement is to assign events to timeslots and resources (corresponding to rooms in our cases), subject to no clashes between resources. Therefore it is believed that the type of decomposition considered in this paper can in principle be applied to other timetabling problems as well.

4. Two-Stage Decomposition of the integer programming model

The TSD of Model (3) is performed as follows. The model is split into two stages; in Stage I, events are assigned to timeslots, and in Stage II, events are assigned to rooms. The respective decision variables for these stages are the following; $y_{e,t} \in \{0,1\}$ takes value 1 if event $e \in \mathcal{E}$ is assigned timeslot $t \in \mathcal{T}$, and 0 otherwise; $z_{e,r} \in \{0,1\}$ takes value 1 if event $e \in \mathcal{E}$ is assigned room $r \in \mathcal{R}$, and 0 otherwise. This means that constraints (3b) and (3c) are part of Stage I, and constraints (3d) and (3e) are part of Stage II.

As for constraints (3f), defined by the set P_{other}, it is assumed that each of the constraints in P_{other} is either only touching the assignment of events to timeslots (denoted P_{timeslot}) or the assignment of events to rooms (denoted P_{room}). It is shown in Section 5 that this assumption holds, with one exception. This means that constraints P_{timeslot} can be fully stated in terms of variable $y_{e,t}$, and constraints P_{room} can be fully stated in terms of variable $z_{e,r}$. As constraints P_{timeslot} are part of Stage I, these are handled optimally. This is different for P_{room}, as those constraints are harder to consider during Stage I. We will therefore address them with greater care in the next sections and show how we can add weighted room allocations to the decomposed model as a good approximation.

The solution obtained from the Stage I model is given as a parameter to the Stage II model, denoted $y^*_{e,t}$. The advantage of this approach is the huge reduction in the number of variables in both stages, which results in a significantly decreased solving time. The following substitution of variables are made:

$$\sum_{r \in \mathcal{R}} x_{e,r,t} = y_{e,t} \quad \text{(Stage I)} \tag{4}$$

$$x_{e,r,t} = y^*_{e,t} z_{e,r} \quad \text{(Stage II)} \tag{5}$$

The objective (3a) of the original model defines a natural objective for both Stage I and Stage II, since it can be split into two independent expressions (denoted w^I and w^{II}, respectively). If this was not the case (e.g. if an event had different priorities for rooms depending on the timeslot it was assigned), it would complicate matters in terms of the Stage I model.

To sum up, Models (6) and (7) show Stage I and Stage II, respectively.

Stage I (6)

$$\min \quad w^I = \sum_{e \in \mathcal{E}, t \in \mathcal{T}} \phi_{e,t} y_{e,t} \tag{6a}$$

s.t.

$$\text{(one timeslot)} \quad \sum_{t \in \mathcal{T}} y_{e,t} = 1 \ \ \forall e \in \mathcal{E} \tag{6b}$$

$$\text{(resource conf.)} \quad \sum_{e \in E'_a} y_{e,t} \le 1 \ \ \forall a \in \mathcal{A}, \ t \in \mathcal{T} \backslash t_D \tag{6c}$$

$$y_{e,t} \in P_{\text{timeslot}} \tag{6d}$$

$$y_{e,t} \in \{0,1\} \tag{6e}$$

Stage II (solution from Stage I is denoted $y^*_{e,t}$) (7)

$$\min \quad w_{II} = \sum_{e \in \mathcal{E}, r \in \mathcal{R}} \pi_{e,r} z_{e,r} \tag{7a}$$

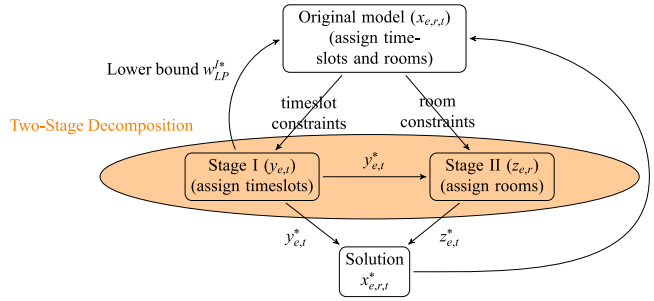


Fig. 1. Two-Stage Decomposition flow chart.

s.t.

$$\text{(one room)} \quad \sum_{r \in \mathcal{R}} z_{e,r} = 1 \ \ \forall e \in \mathcal{E} \tag{7b}$$

$$\text{(room conf.)} \quad \sum_{r \in \mathcal{E}} y^*_{e,t} z_{e,r} \le G_{r,t} \ \ \forall r \in \mathcal{R} \backslash r_D, \ \ r \in \mathcal{T} \backslash t_D \tag{7c}$$

$$\text{(eligible rooms)} \quad z_{e,r} \le K_{e,r} \ \ \forall e \in \mathcal{E}, r \in \mathcal{R} \tag{7d}$$

$$z_{e,r} \in P_{\text{room}} \tag{7e}$$

$$z_{e,r} \in \{0,1\} \tag{7f}$$

The outline of the TSD is shown in Fig. 1. For a variable x the star-suffixed version x^* denotes a feasible solution. Stage I is solved using a MIP solver to obtain a solution $y^*_{e,t}$, which is given as input to the Stage II model. Note that Stage I possesses the coloring structure from Theorem 3.1. Therefore Stage I is already a hard problem in its most basic form. Furthermore the value of the LP-relaxation of the Stage I model (denoted w^{I*}_{LP}) is a lower bound on the original model, as the Stage I model can be seen as a relaxation. Solving the Stage II model subject to the solution of the Stage I model obtains a solution $z^*_{e,t}$, and a solution to the original model $x^*_{e,r,t}$ can then be derived by Eqs. (4) and (5).

For this paper we will solve Stage II using the specified IP even though we have not established its complexity yet. But as constructing a polynomial time algorithm for Stage II that can cope with all the additional constraints would be out of this papers scope we will postpone this to a potentially later point in time. In the computational results section we will see that solving Stage II will not be the time-wise bottleneck anyhow.

As previously discussed, the penalties for room allocation can be implicitly handled in Stage I, which is described in Section 4.1. This extension of Stage I will not only allow better solutions to be found, but possibly also improvements in the bounds found by means of the LP relaxation.

As an alternative approach, we remark that an iterative procedure could in principle be used, such that the solution obtained from the Stage II model is given as input to the Stage I model, and the whole procedure is repeated. It is however unclear how the input from the Stage II model should effect the Stage I model to obtain convergence towards better solutions in terms of the overall objective. Furthermore, such an approach would require that both Stage I and Stage II can be solved 'quickly' (for the practical problem treated in this paper, computational results will show that this is in fact not the case for the Stage I model).

4.1. Extending Stage I with room allocation

The key idea behind extending Stage I with room allocation penalties is to consider Stage II as a matching problem in a bipartite graph. Constraints (7e) are set aside in the following, as they have not been defined yet. However, it will be seen later that

these constraints do not fully obey the matching problem structure, and therefore Stage II must be solved with a MIP solver. This means that the room penalties are only partly incorporated in Stage I, but still this seems better than having Stage I being totally unaware of these penalties, as already discussed in Section 1.

Some basic graph notation is introduced in the following. A graph is *bipartite* if its set of vertices can be partitioned into two sets A and B, such that every edge has one endpoint in A, and the other endpoint in B. A *matching* in a graph is a set of edges such that no two of these edges share endpoints. A *maximum matching* is a matching that contains the largest possible number of edges. The *matching number* $\nu(G)$ of graph G is the number of edges in a maximum matching. For a graph with edge-weights, a *minimum weighted maximum matching* is a maximum matching where the sum of the weights on the edges of the matching is minimal.

In the Stage II model (7), notice first that the only constraint which treats timeslots is constraint (7c). Since this constraint applies to timeslots individually, Model (7) can be split into $|\mathcal{T}|$ independent optimization problems. Second, assume that the minimum weighted maximum matching problem of the weighted bipartite graph $G_t=(\mathcal{E}\cup\overline{\mathcal{R}},E_t)$ fully describes the optimization problem of timeslot t of Model (7). To recognize this, let $\mathcal{R}_D$ be the set of $|\mathcal{E}|$ distinct dummy-rooms. i.e. For each event a dummy-room is created (and a corresponding edge is added to the graph) to ensure a matching of every event to a room will always exist. Hence the room-vertices of graph G_t is given by $\overline{\mathcal{R}}=\mathcal{R}\cup\mathcal{R}_D$. The set of edges is given by (skipping edge definitions for the dummy-room vertices) $E_t=\{e\in\mathcal{E},r\in\mathcal{R}|K_{e,r}=1\wedge G_{r,t}=1\}$, and the weight on each edge is given by $\pi_{e,r}$. The goal of the matching problem is to select a maximum matching with minimum weight. A trivial maximum matching will assign every event to the dummy-room. Clearly this resembles component $t\in\mathcal{T}$ of Model (7).

Stating the Stage II model in terms of this graph allows us to exploit some well-known properties of matching problems in bipartite graphs. In the following, notation is simplified by dropping the t-index where applicable, i.e. we write G instead of G_t and E instead of E_t. Denote by $\Gamma(S)$ the neighbors of event-nodes $S\subseteq\mathcal{E}$ in graph G, i.e. $\Gamma(S)=\{i\in\overline{\mathcal{R}}|j\in S,(i,j)\in E\}$. Hence $\Gamma(S)\subseteq\overline{\mathcal{R}}$. The well-known theorem of Hall states that a bipartite graph $G=(\mathcal{E}\cup\overline{\mathcal{R}},E)$ has a matching of all vertices $\mathcal{E}$ into $\overline{\mathcal{R}}$ if and only if $|\Gamma(S)|\geq|S|\ \forall S\subseteq\mathcal{E}$. Observe that for timeslot $t\in\mathcal{T}$, the variable $y_{e,t}$ determines whether event $e\in\mathcal{E}$ is part of graph G. Lach and Lübbecke [18] used this theorem to add constraints of the form

$$\sum_{e\in S}y_{e,t}\leq|\Gamma(S)|\quad\forall S\subseteq\mathcal{E},t\in\mathcal{T}\tag{8}$$

to the Stage I model to guarantee that the Stage I model would yield a feasible matching problem for every component $t\in\mathcal{T}$ of the Stage II model. However, such constraints are redundant in our case, as we are guaranteeing that no matter how $y_{e,t}$ is selected, a feasible matching will always exist (due to the dummy-rooms). Instead we modify the expression (8) to provide a lower bound on the weighted matching problem.

For the bipartite graph $G=(\mathcal{E}\cup\overline{\mathcal{R}},E)$ (the edge-weights are set aside for now), let the *deficiency* of a vertex set $S\subseteq\mathcal{E}$ be defined as $\mathrm{def}(S)=|S|-|\Gamma(S)|$. Let the deficiency of G be defined as $\mathrm{def}(G)=\max_{S\subseteq\mathcal{E}}\mathrm{def}(S)$. Theorem 1.3.1 of Lovász and Plummer [20] states the following:

Theorem 4.1. *The matching number of the bipartite graph* $G=(\mathcal{E}\cup\overline{\mathcal{R}},E)$, *is* $\nu(G)=|\mathcal{E}|-\mathrm{def}(G)$.

i.e. for the bipartite graph G, $\mathrm{def}(G)$ denotes the amount of vertices which are not matched in the maximum matching.

Let $\mathcal{W}$ denote the ordered set of different values found in $\pi_{e,r}$, i.e.

$$\mathcal{W}=\{w\in\mathbb{R}^+|\exists e\in\mathcal{E},\ \exists r\in\mathcal{R}:\pi_{e,r}=w\}\tag{9}$$

$$w_i<w_j\Leftrightarrow\mathrm{ord}w_i<\mathrm{ord}w_j\quad\forall(w_i,w_j)\in\mathcal{W}\tag{10}$$

Notice that no restrictions are posed on the amount of different values found, but it should be remarked that for our practical case, the cardinality of $\mathcal{W}$ is small (typically below 10).

The bipartite graph G is split into subgraphs, one subgraph for each $w\in\mathcal{W}$. A subgraph is denoted as $G_{\leq w}=(\mathcal{E}\cup\overline{\mathcal{R}},E_{\leq w})$, where the set of edges are those with at least weight w,

$$E_{\leq w}=\{(e,r)\in E|\pi_{e,r}\leq w\}\tag{11}$$

By these definitions, it is clear that

$$|\Gamma(G_{\leq w_1})|\leq|\Gamma(G_{\leq w_2})|\leq\ldots\Rightarrow\tag{12}$$

$$\mathrm{def}(G_{\leq w_1})\geq\mathrm{def}(G_{\leq w_2})\geq\ldots\tag{13}$$

Using the deficiencies of these subgraphs, a lower bound on the minimum weight maximum matching can be stated. Let $a_w\in\mathbb{N}_0$ be defined as

$$a_{w_i}=\begin{cases}\nu(G_{\leq w_i})-\nu(G_{\leq w_{i-1}})=\mathrm{def}(G_{\leq w_{i-1}})-\mathrm{def}(G_{\leq w_i}) & \text{if } i>1\\ \nu(G_{\leq w_1})=|\mathcal{E}|-\mathrm{def}(G_{\leq w_1}) & \text{otherwise}\end{cases}\tag{14}$$

The intuition behind a_w is to measure the change in the matching number when edges with weight w are added to the subgraph $G_{\leq w_{i-1}}$. Note that $0\leq a_w\leq|\mathcal{E}|$ for any $w\in\mathcal{W}$, as $0\leq\mathrm{def}(G_{\leq w})\leq|\mathcal{E}|$.

Theorem 4.2. *The quantity*

$$\sum_{w\in\mathcal{W}}w\cdot a_w$$

is a lower bound on a minimum weight maximum matching in the edge-weighted bipartite graph G.

Proof. Assume for contradiction there exists a maximum matching M with lower weight, i.e.

$$\sum_{e\in M}w_e<\sum_{w\in\mathcal{W}}w\cdot a_w$$

Let b_w denote the number of edges in M of weight lesser or equal w, i.e.

$$b_w=|\{e\in M:w_e\leq w\}|$$

Let k be the smallest number such that,

$$b_{w_k}>\sum_{i=1}^{k}a_{w_k}$$

This number must exist since M is a cheaper matching. For the subgraph $G_{\leq w_k}$, b_{w_k} can never exceed the matching number $\nu(G_{\leq w_k})$ (by Theorem 4.1). We say 'exceed' as the matching might not include precisely $\nu(G_{\leq w_k})$ edges of weight lesser or equal w_k. This gives

$$\begin{aligned}b_{w_k}&\leq\nu(G_{\leq w_k})\\&=|\mathcal{E}|-\mathrm{def}(G_{\leq w_k})\\&=|\mathcal{E}|\underbrace{-\mathrm{def}(G_{\leq w_1})+\mathrm{def}(G_{\leq w_1})-\mathrm{def}(G_{\leq w_2})+\mathrm{def}(G_{\leq w_2})\ldots+\mathrm{def}(G_{\leq w_{k-1}})}_{=0}\\&\quad-\mathrm{def}(G_{\leq w_k})\\&=\sum_{i=1}^{k}a_{w_i}\end{aligned}$$

which is a contradiction. □

This lower bound is minimized in the objective of the Stage I model. Hence, any lower bound on the Stage I model is a lower bound on the overall problem. Additional notation is needed for stating the extended Stage I model.

The neighbors of event-nodes $S\subseteq\mathcal{E}$ in graph $G_{t,\leq w}$ are denoted $\Gamma_{t,\leq w}(S)$ for timeslot t and weight w. Let the variable $\mathrm{def}_{t,\leq w}\in\mathbb{N}_0$

be the deficiency of subgraph $G_{t,\le w}$. The deficiencies for each subgraph can be determined by adding the following constraint (follows directly from Theorem 4.1 and the definition of the deficiency for a bipartite graph),

$$\sum_{e\in S} y_{e,t} - \text{def}_{t,\le w} \le |\Gamma_{t,\le w}(S)| \quad \forall S\subseteq\mathcal{E},\ t\in\mathcal{T},\ w\in\mathcal{W} \tag{15}$$

Model (16) shows the extended model. Variables $\text{def}_{t,\le w}$ and $a_{t,w}$ are specified to be continuous as they will naturally take integer values. Obviously an exponential amount of constraints is added due to (16d), but it will be shown that for our practical purpose, the amount of required constraints is low.

Stage I extended with Hall's condition (16)

$$\min \quad W^I = \sum_{e\in\mathcal{E},t\in\mathcal{T}} \phi_{e,t} y_{e,t} + \sum_{t\in\mathcal{T},w\in\mathcal{W}} w a_{t,w} \tag{16a}$$

s.t.

$$\text{(one timeslot)} \quad \sum_{t\in\mathcal{T}} y_{e,t} = 1 \quad \forall e\in\mathcal{E} \tag{16b}$$

$$\text{(resource conf.)} \quad \sum_{e\in E_a'} y_{e,t} \le 1 \quad \forall a\in\mathcal{A},\ t\in\mathcal{T}\backslash t_D \tag{16c}$$

$$\text{(Hall's)} \quad \sum_{e\in S} y_{e,t} - \text{def}_{t,\le w} \le |\Gamma_{t,\le w}(S)| \forall S\subseteq\mathcal{E},\ t\in\mathcal{T},\ w\in\mathcal{W} \tag{16d}$$

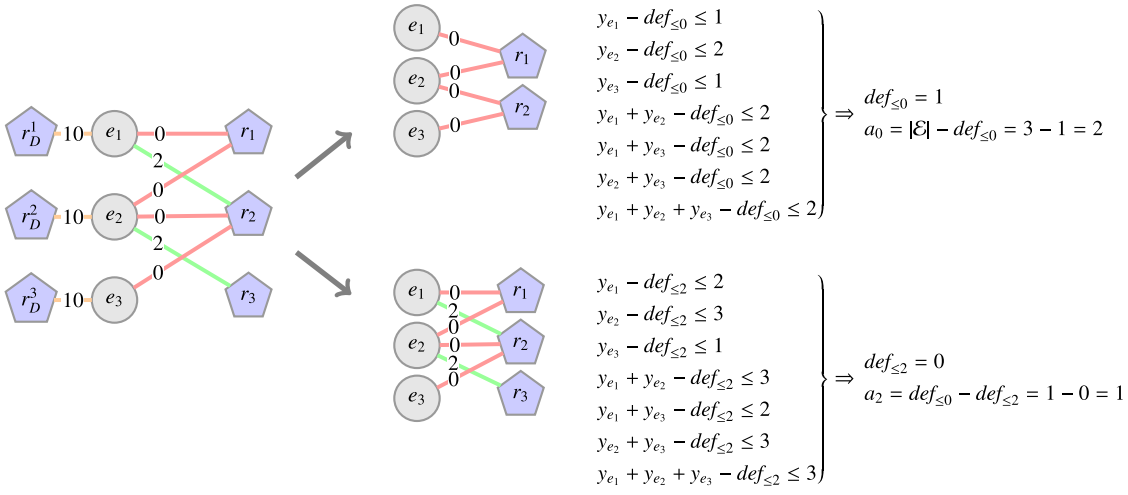
$$\text{(LB)} \quad |\mathcal{E}| - \text{def}_{t,\le w_1} = a_{t,w_1} \quad \forall t\in\mathcal{T} \tag{16e}$$

$$\text{(LB)} \quad \text{def}_{t,\le w_{-1}} - \text{def}_{t,\le w} = a_{t,w} \quad \forall t\in\mathcal{T},\ w\in\mathcal{W},\ \text{ord}(w)>1 \tag{16f}$$

$$y_{e,t}\in\{0,1\} \tag{16g}$$

$$\text{def}_{t,\le w},\ a_{t,w}\in\mathbb{R}^+ \tag{16h}$$

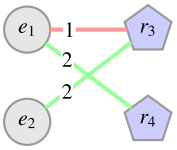
Example 4.1. Below is shown an example of a bipartite graph and its subgraphs for some timeslot. Three different room-weights exist, $\mathcal{W}=\{0,2,10\}$. Clearly, an optimal solution to the matching problem of this graph is $(e_1,r_1),(e_2,r_3),(e_3,r_2)$ with value 2. The subgraphs for weights 0 and 2 are shown, and the lower bound derived.



Hence the lower bound is derived as

$$LB = a_0 0 + a_2 2 = 2$$

Example 4.2. Naturally, the lower bound is not necessarily tight, as shown by the following small example.



For weight $w_1=1$ the deficiency of G_1 is $\text{def}(G_1)=1$ and therefore $a_1=1$. As the deficiency for G_2 is $\text{def}(G_2)=0$ we also have $a_2=1$. The lower bound therefore reads $1\cdot 1+2\cdot 1=3$. But obviously the only (and therefore minimal weight) maximum matching has weight 4. In fact, by increasing the weight on the weight 2 edges, it is seen that the gap between the lower bound and the actual minimal weight maximum matching could potentially be arbitrarily large. For the practical problem handled later, the weights can take values $\{1,2,\ldots,10\}$, and therefore the gap between weights is low. The gap between the lower bound and the actual matchings obtained will be investigated experimentally.

4.2. An exact approach using Egerváry's theorem

An alternative approach to the derived lower bound, the theorem of Egerváry [15] can be used to characterize the minimum weight of a matching in a bipartite graph, which deserves a mentioning in this context. The theorem states the following ([30, Theorem 17.1], here stated as a minimum weight problem):

Theorem 4.3. *Let $G=(V,E)$ be a bipartite graph and let $w: E\to\mathbb{R}^+$ be a weight function. Then the minimum weight of a matching in G is equal to the maximum value of $y(V)$, where $y: V\to\mathbb{R}^+$ is such that*

$$y_u + y_v \le w_e \quad \forall u,v\in V,(u,v)\in E$$

However, since we consider a bipartite graph for each timeslot, and each bipartite graph in worst case has $|\mathcal{E}||\mathcal{R}|$ vertices (which occurs often in practice), the amount of required constraints is of magnitude $|\mathcal{E}||\mathcal{R}||\mathcal{T}|$, so this is not a tractable approach. Furthermore, since the graph is not static (i.e. its structure depends on assignments of events to timeslots), a min–max formulation would be required.

4.3. Generating Hall's inequalities

To generate the Hall's inequalities, it is necessary to exploit the structure of the underlying graph. Thereby we use problem specific knowledge to overcome the requirement of enumerating all subsets of events. In the Lectio case, two important features are known about the bipartite graphs:

- An event often has a special association with one specific room. This is either because the event is locked to that room, or because a penalty is imposed on *not* assigning an event to the room it was previously assigned to. In the later case, this means that one room has a lower weight than all other rooms for the particular event. Hence, in the subgraph $G_{\leq w}$ for this respective lower weight, only a single edge exists for the event. An event with only a single adjacent edge is denoted as a *singleton* event from now on.
- Furthermore, predefined feature-groups of rooms exist. A feature group of rooms is devoted to a certain type of lecture, for instance chemistry or physics, which require a room with specialized equipment. Hence many events are adjacent to the exact same set of rooms.

These graphs are hence exploited by separately considering the inequalities induced by singleton events and the inequalities induced by all other events, and finally those inequalities induced by combining these. The approach is formalized below. It should be remarked that applying this type of decomposition will require exploiting at least some properties of the underlying graph. We refer to Balas and Pulleyblank [3], Edmonds [14], Lach and Lübbecke [17] as helpful resources in this aspect.

For a subset of rooms $R \subseteq \mathcal{R}$, let $\Gamma^{-1}(R)$ be the set of events adjacent to *only* rooms in R, i.e. $\Gamma^{-1}(R) = \{e \in \mathcal{E} | \Gamma(\{e\}) \subseteq R\}$.

Theorem 4.4. *The Hall inequalities*

$$\sum_{e \in S} y_e - def \leq |\Gamma(S)| \qquad \forall S \subseteq \mathcal{E}$$

are fully contained in

$$\sum_{e \in \Gamma^{-1}(R)} y_e - def \leq |R| \qquad \forall R \subseteq \mathcal{R}$$

Proof. Let $S \subseteq \mathcal{E}$ be any set of events. Now we let $R = \Gamma(S)$. Obviously we have $S \subseteq \Gamma^{-1}(R) = \Gamma^{-1}(\Gamma(S))$. If $\sum_{e \in \Gamma^{-1}(R)} y_e - \text{def} \leq |R|$ holds we get

$$\sum_{e \in S} y_e - \text{def} \leq \sum_{e \in \Gamma^{-1}(R} y_e - \text{def} \leq |R| = |\Gamma(S)| \qquad \square$$

This means that instead of having a constraint for every subset of events we can do with a constraint for every subset of rooms (which are considerably less).

Next we can further reduce the number of necessary constraints by exploiting symmetry between rooms. Rooms which are adjacent to exactly the same events can be grouped, and essentially treated as one room.

Theorem 4.5. *Let* $R_1, \ldots, R_m \subseteq \mathcal{R}$ *be distinct* $(i \neq j \Rightarrow R_i \cap R_j = \varnothing)$ *subsets of rooms. Let* $I \subseteq \{1, \ldots, m\}$ *be an index set such that*

$$\bigcup_{i \in I} \Gamma^{-1}(R_i) = \Gamma^{-1}\left(\bigcup_{i \in I} R_i\right) \text{(i.e. there is no event that only fits into a combination of the room sets in } I\text{)}$$

Then the Hall constraint

$$\sum_{e \in \Gamma^{-1}(\bigcup_{i \in I} R_i)} y_e - \text{def} \leq \left|\bigcup_{i \in I} R_i\right|$$

is dominated by

$$\sum_{e \in R_i} y_e - \text{def}_i \leq |R_i| \qquad \forall i \in I$$
$$\sum_{i \in I} \text{def}_i \quad \leq \text{def}$$

where $\text{def}_i \in \mathbb{N}_0$ *is the deficiency of index* $i \in I$, *i.e.* $\text{def}_i = \text{def}(\Gamma^{-1}(R_i))$.

Proof. First note that $\sum_{i \in I} \text{def}_i \leq \text{def}$ implies

$$\sum_{e \in \Gamma^{-1}(\bigcup_{i \in I} R_i)} y_e - \text{def} \leq \sum_{i \in I} \sum_{e \in \Gamma^{-1}(R_i)} y_e - \text{def}_i$$

Next by $\sum_{e \in \Gamma^{-1}(R_i)} y_e - \text{def}_i \leq |R_i|$ we get

$$\sum_{i \in I} \sum_{e \in \Gamma^{-1}(R_i)} y_e - \text{def}_i \leq \sum_{i \in I} |R_i| = \left|\bigcup_{i \in I} R_i\right|$$

where the later equality holds as the R_i are distinct. $\square$

Since the amount of possible ways to select I is exponential, this shows a potential way to limit the amount of necessary inequalities.

The graphs of the Lectio instances usually have the following structure, as previously discussed: Certain events are fixed to a specific room. These events are known as singleton events, and are denoted with the set E^1. If the singleton events are discarded, all other rooms can be grouped into groups $\mathcal{I} = \{1, 2, \ldots, m\}$, i.e. $R_i \subseteq \mathcal{R}\ \forall i \in \mathcal{I}$, where every room is connected to the very same events as the other rooms of the same group. In particular this means

$$\Gamma^{-1}(R) \backslash E^1 = \varnothing \quad \forall R \subsetneq R_i, i \in \mathcal{I} \tag{17}$$

meaning that no event is adjacent to only a subset of rooms of the room-groups, except for the singleton events. The key observation here is that the number of these groups of rooms is low, yielding a tractable way to generate the Hall inequalities. By Theorem 4.4 we know that a subset of rooms fully characterises one of the Hall constraints (and that it is sufficient to consider only those).

Corollary 4.1. *Given the structure of the Lectio graphs, only the following subsets of rooms need to be considered w.r.t. Eq.* (15) (*in the altered form defined by Theorem* 4.4):

$$(I) \quad \Gamma(e) \quad \forall e \in E^1 \tag{18}$$

$$(II) \quad \bigcup_{i \in I} R_i \quad \forall I \subseteq \mathcal{I} \tag{19}$$

Proof. For contradiction, let $\tilde{R} \subseteq \mathcal{R}$ be any other subset of rooms, i.e.

$$\tilde{R} \neq \Gamma(e) \quad \forall e \in E^1$$
$$\tilde{R} \neq \bigcup_{i \in I} R_i \quad \forall I \subseteq \mathcal{I}$$

$\tilde{R}$ can be decomposed into subsets $\tilde{R}_i, i \in \mathcal{I}$, such that $\tilde{R}_1 \subseteq R_1$, $\tilde{R}_2 \subseteq R_2, \ldots, \tilde{R}_m \subseteq R_m$ and $\bigcup_{i \in \mathcal{I}} \tilde{R}_i = \tilde{R}$.

For each of the decomposed room sets $\tilde{R}_i$ we can now have one of the three following cases (by Eq. (17), which disallows that $\tilde{R}_i \neq R_i$ and $\Gamma^{-1}(\tilde{R}_i) \backslash E^1 \neq \varnothing$):

1. $\tilde{R}_i = R_i$
2. $\tilde{R}_i \neq R_i$ and $\Gamma^{-1}(\tilde{R}_i) \cap E^1 \neq \varnothing$
3. $\tilde{R}_i \neq R_i$ and $\Gamma^{-1}\tilde{R}_i = \varnothing$

Let the respective indices be contained in the sets I_1, I_2 and I_3. The rooms from the third case $(\tilde{R}_i, i \in I_3)$ do not add events to the left hand side of a Hall constraint and can therefore be ignored.

If combining the rooms from the first case to $R' = \bigcup_{i \in I_1} \tilde{R}_i$ we get one of the already considered combinations of room groups. Now note that there is no event fitting into the combination of R' with any of the rooms from the second case (their Γ^{-1} only contains singleton events) and therefore the condition for Theorem 4.5 is met

$$\left(\bigcup_{i \in I_2} \Gamma^{-1}(\tilde{R}_i)\right) \cup \Gamma^{-1}(R') = \Gamma^{-1}\left(\left(\bigcup_{i \in I_2} \tilde{R}_i\right) \cup R'\right)$$

So we now know that the Hall constraint corresponding to $\tilde{R}$ is unnecessary. □

Algorithm 4.1 shows the implemented algorithm for generation all necessary Hall constraints according to this construction.

Algorithm 4.1. Generating Hall's conditions.

1: **input**: bipartite graph $G = (\mathcal{E} \cup \overline{\mathcal{R}}, E)$
2: **output**: set of rooms H, which each constitute a Hall inequality
3: identify E^1 of G
4: $N_r = \{e \in \mathcal{E} \backslash E^1 | r \in \Gamma(e) \neq \emptyset\}$ ▷ Identity adjacent events for each room
5: $T = \{R \subseteq \overline{\mathcal{R}} | (r_i, r_j) \in R, i \neq j, N_{r_i} = N_{r_j}\}$ ▷ Groups of rooms which are adjacent to the same events
6: **for all** $S \in \mathcal{P}(T)$ **do** ▷ $\mathcal{P}(T)$ denotes the powerset of T, i.e. the collection of all subsets
7: $H = H \cup \{r \in R | R \in S\}$ ▷ Add set of room (Eq. (19))
8: **end for**
9: $H = H \cup \{r\} \quad \forall r \in \Gamma(E^1)$ ▷ Add rooms of singleton events (Eq. (18))

In Line 5 rooms are grouped. Here it should be remarked that this is done in a way that identifies the minimum number of groups of rooms. The amount of generated inequalities is hence exponential in the number of groups of rooms. For the Lectio high school timetabling problem, this number is in general low. However, an artificial limit of a maximum of 12 different groups of rooms is imposed, allowing in magnitude of 2^{12} inequalities to be generated for each timeslot. In practice, only two problem instances are restricted by this limit ("HasserG2012" and "SlagelG2012"). The room groups to generate inequalities are selected by ordering the room groups in terms of total number of adjacent events to all other room groups, and taking those room groups where this number is highest. Obviously, omitting some inequalities will not change the fact that the room allocation penalty added to Stage I is a lower bound on the objective of Stage II.

5. Lectio high school timetabling problem

To establish computational results, Stage I and Stage II are extended to the full version of the Danish case of high school timetabling described in Sørensen and Stidsen [31]. This variant of the problem is used in the timetabling component of the high school ERP-system Lectio, and hence reflects all aspects of a practical timetabling optimization problem. Lectio Timetabling is used by many high schools in Denmark, and this formulation of the problem has been used in production mode for over a year. In this paper a brief introduction to each of the added constraints and variables is given. More in-depth description and motivation can be found in Sørensen and Stidsen [31].

This timetabling problem contains more types of constraints than what is usually found in the literature. This is mainly related to the big number of different high schools which use it, which inevitably gives a big variety of required features. However, a conversion scheme from this timetabling problem to the general XHSTT format is known, so the Lectio problem fits within the general concepts of high school timetabling problems.

Extensive computational experiments have shown that the usual formulation of this timetabling problem using a binary variable with three indices is very challenging for the commercial MIP solver Gurobi 5, which is among the very best general-purpose MIP solvers according to recent benchmarks of Mittelman [22]. Therefore this timetabling problem is a good candidate for testing the TSD approach.

5.1. Stage I

The set of timeslots $\mathcal{T}$ is defined by the combination of the set of days $\mathcal{D}$, and the set of daily-timeslots (known as *modules*) $\mathcal{M}$. The set of resources $\mathcal{A}$ consists of teachers and students, which is also known as the set of *entities*. Furthermore, the set of *classes* is denoted $\mathcal{C}$. A class $c \in \mathcal{C}$ is a *non-physical* resource treating a specific teaching-subject, and is associated with a certain set of events. Hence, an event can be viewed as a single lecture of a certain class. Parameter $J_{e,c} \in \{0,1\}$ takes value 1 if event $e \in \mathcal{E}$ is part of class $c \in \mathcal{C}$, and 0 otherwise.

Variable $\nu_{a,t} \in \{0,1\}$ takes value 1 if entity $a \in \mathcal{A}$ is active in timeslot $t \in \mathcal{T}$, and 0 otherwise. Variable $f_{a,d} \in \{0,1\}$ takes value 1 if entity $a \in \mathcal{A}$ has no events scheduled on events on day $d \in \mathcal{D}$ (we say that the entity has a *day off*, even though he/she might be occupied by unscheduled activities, such as lecture preparation), and 0 otherwise. Variable $b_{c,t} \in \{0,1\}$ takes value 1 if class $c \in \mathcal{C}$ has at least one lecture in timeslot $t \in \mathcal{T}$, and 0 otherwise. Variable $n_{c,d} \in \{0,1\}$ takes value if class $c \in \mathcal{C}$ has a neighbor-day conflict on day $d \in \mathcal{D}$. A neighbor-day conflict occurs when the same class has scheduled events on two consecutive days. Variable $o_{a,d} \in \{0,1\}$ takes value 1 if entity $a \in \mathcal{A}$ has only one event on day $d \in \mathcal{D}$, and 0 otherwise. Days with only one lecture are undesirable and should be avoided. Variable $w_c \in \mathbb{N}_0$ is the amount which class $c \in \mathcal{C}$ is 'out of week-balance'. i.e. If the set of timeslots is made up of times from more than one week, the amount of events of each class in each week must be equivalent (as far as possible). Variable $h_{a,d} \in \mathbb{N}_0$ is the amount of idle timeslots (a timeslot with no activity, but there exists both at least one earlier and one later timeslot with activity) for entity $a \in \mathcal{A}$ on day $d \in \mathcal{D}$. Variables $\underline{h}_{a,d}$, $\overline{h}_{a,d} \in \mathbb{N}_0$ denote the ordinal number of the first and last timeslot with activity on day $d \in \mathcal{D}$ for entity $a \in \mathcal{A}$, respectively.

The objective consists of 6 additional terms. These denote the weighted sum of entity idle slots (weight $\phi_a \in \mathbb{R}^+$), neighbor-day conflicts (weight $\zeta \in \mathbb{R}^+$), days with only one lecture (weight $\eta_a \in \mathbb{R}^+$), days-off for teachers (weight $\gamma_a \in \mathbb{R}^+$), days-off for students (weight $\delta_a \in \mathbb{R}^+$), and class week stability (weight $t \in \mathbb{R}^+$), respectively.

Let parameters S_e and C_e be the set of events which should be scheduled in the same timeslot as event $e \in \mathcal{E}$, and in the timeslot immediately following event $e \in \mathcal{E}$, respectively. Parameter $P_{d,d'} \in \{0,1\}$ takes value 1 if day $d \in \mathcal{D}$ and day $d' \in \mathcal{D}$ are neighbor-days, and 0 otherwise. Parameter $R_{c,d} \in \{0,1\}$ takes value 1 if class $c \in \mathcal{C}$ is part of some event which is locked to some timeslot on day $d \in \mathcal{D}$, and let $N_c \in \mathbb{N}_0$ be the number of allowed neighbor-day conflicts for class $c \in \mathcal{C}$. $\mathcal{T}_d$ denotes the set of timeslots on day $d \in \mathcal{D}$. Parameter $D_{e,t} \in \{0,1\}$ takes value 1 if event $e \in \mathcal{E}$ can be scheduled in timeslot $t \in \mathcal{T}$, and 0 otherwise. Parameter $F_a \in \mathbb{N}_0$ denotes the amount of required days off for entity $a \in \mathcal{A}$. Parameter $W_a \in \mathbb{N}_0$ denotes the maximum amount of events which can be scheduled to entity $a \in \mathcal{A}$ on any given day.

A class can only have one event assigned to each day, unless it is specified that multiple events should be placed in contiguous positions. We say that such *day-conflicts* are infeasible. The set $E''' \subseteq \mathcal{E}$ denotes the set of events for which day-conflicts are checked.

The most common case is that a school creates a timetable for a single week. However, some schools desire to create a two-week timetable instead. This allows more flexibility in the planning; take for instance a class with a nominated teaching load of three events per week. In case the school uses a two-week timetable, this class can for instance have one double lecture in the first week, and two double lectures in the second week. $d(\underline{\mathcal{T}})$ and $d(\overline{\mathcal{T}})$ denote the sets of days in the first and in the second week, respectively. $\underline{\mathcal{T}}$ and $\overline{\mathcal{T}}$ denote the timeslots in the first and second week, respectively.

The complete Stage I model is shown in (20).

Stage I Lectio (20)

$$\min w^I = \sum_{e\in\mathcal{E},\ t\in\mathcal{T}} \phi_{e,t} y_{e,t} + \sum_{t\in\mathcal{T},\ w\in\mathcal{W}} w a_{t,w} + \sum_{a\in\mathcal{A},\ d\in\mathcal{D}} \beta_a h_{a,d} + \zeta \sum_{c\in\mathcal{C},\ d\in\mathcal{D}} n_{c,d} + \sum_{a\in\mathcal{A},\ d\in\mathcal{D}} \eta_a o_{a,d} + \sum_{a\in\mathcal{A}} \gamma_a \left[|\mathcal{D}| - \sum_{d\in\mathcal{D}} f_{a,d} \right] + \sum_{a\in\mathcal{A}, d\in\mathcal{D}} \delta_a f_{a,d} + \iota \sum_{c\in\mathcal{C}} w_c \quad (20a)$$

s.t.

$$\text{(one timeslot)} \sum_{t\in\mathcal{T}} y_{e,t} = 1 \quad \forall e\in\mathcal{E} \quad (20b)$$

$$\text{(entity time aux.)} \sum_{e\in E_a} y_{e,t} = v_{a,t} \quad \forall a\in\mathcal{A}, t\in\mathcal{T} \quad (20c)$$

$$\text{(entity conf.)} \sum_{t\in\mathcal{T}_d} v_{a,t} + f_{a,d} \le 1 \quad \forall a\in\mathcal{A}, d\in\mathcal{D} \quad (20d)$$

$$\text{(Hall's)} \sum_{e\in S} y_{e,t} - \text{def}_{t,\le w} \le |\Gamma_{t,\le w}(S)| \forall S\subset\mathcal{E}, t\in\mathcal{T}, w\in\mathcal{W} \quad (20e)$$

$$\text{(room alloc. lb)} |\mathcal{E}| - \text{def}_{t,\le w_1} = a_{t,w_1} \quad \forall t\in\mathcal{T} \quad (20f)$$

$$\text{(room alloc. lb)} \text{def}_{t,\le w_{-1}} - \text{def}_{t,\le w} = a_{t,w} \quad \forall t\in\mathcal{T}, w\in\mathcal{W},\ \text{ord}(w)>1 \quad (20g)$$

$$\text{(locked time)} y_{e,t} = 1 \quad \forall e\in\mathcal{E}, t\in\mathcal{T}, LT_{e,t}=1 \quad (20h)$$

$$\text{(same time)} y_{e,t} - y_{e't} = 0 \quad \forall e\in\mathcal{E}, e'\in S_e, t\in\mathcal{T} \quad (20i)$$

$$\text{(cont. times)} y_{e,t} - y_{e't'} = 0 \quad \forall e\in\mathcal{E}, e'\in C_e, (t,t')\in\mathcal{T}, d_t = d_{t'}, \text{ord}(t)+1 = \text{ord}(t') \quad (20j)$$

$$\text{(n.d. conf.)} \sum_{t\in\mathcal{T}_d} b_{c,t} + \sum_{t\in\mathcal{T}_{d'}} b_{c,t} - n_{c,d} \le 1 \quad \forall c\in\mathcal{C}, (d,d')\in\mathcal{D}, P_{d,d'}=1, R_{c,d}+R_{c,d'} \le 1 \quad (20k)$$

$$\text{(n.d. conf.)} \sum_{d\in\mathcal{D}} n_{c,d} \le N_c \quad \forall c\in\mathcal{C} \quad (20l)$$

$$\text{(forbid. times)} \sum_{t\in\mathcal{T},\ D_{e,t}=0} y_{e,t} = 0 \quad \forall e\in\mathcal{E} \quad (20m)$$

$$\text{(idle slots)} \overline{h}_{a,d} - \underline{h}_{a,d} - \sum_{t\in\mathcal{T}_d} v_{a,t} + 1 = h_{a,d} \quad \forall a\in\mathcal{A}, d\in\mathcal{D} \quad (20n)$$

$$\text{(idle slots)} |\mathcal{M}| - (|\mathcal{M}| - \text{ord}(t)) v_{a,t} \ge \underline{h}_{a,d} \quad \forall a\in\mathcal{A}, d\in\mathcal{D}, t\in\mathcal{T}_d \quad (20o)$$

$$\text{(idle slots)} \text{ord}(t) v_{a,t} \le h_{a,d} \quad \forall a\in\mathcal{A}, d\in\mathcal{D}, t\in\mathcal{T}_d \quad (20p)$$

$$\text{(days off)} \sum_{d\in\mathcal{D}} f_{a,d} \ge F_a \quad \forall a\in\mathcal{A} \quad (20q)$$

$$\text{(days off)} \sum_{t\in\mathcal{T}_d} v_{a,t} + f_{a,d} \ge 1 \quad \forall a\in\mathcal{A}, d\in\mathcal{D} \quad (20r)$$

$$\text{(day conf.)} \sum_{e\in E'''} J_{e,c} y_{e,t} \le b_{c,t} \quad \forall c\in\mathcal{C}, t\in\mathcal{T} \quad (20s)$$

$$\text{(day conf.)} \sum_{t\in\mathcal{T}_d}^{e\in E'''} b_{c,t} \le 1 \quad \forall c\in\mathcal{C}, d\in\mathcal{D} \quad (20t)$$

$$\text{(work limit)} \sum_{e\in\mathcal{E},\ t\in\mathcal{T}_d} y_{e,t} \le W_a \quad \forall a\in\mathcal{A}, d\in\mathcal{D} \quad (20u)$$

$$\text{(one lecture)} 2 - \sum_{t\in\mathcal{T}_d} v_{a,t} - 2 f_{a,d} \le o_{a,d} \quad \forall a\in\mathcal{A}, d\in\mathcal{D} \quad (20v)$$

$$\text{(class stabl.)} \left| \sum_{e\in\mathcal{E},\ t\in\underline{\mathcal{T}}} J_{e,c} y_{e,t} - \sum_{e\in\mathcal{E},\ t\in\overline{\mathcal{T}}} J_{e,c} y_{e,t} \right| - 1 = w_c \quad \forall c\in\mathcal{C} \quad (20w)$$

$$\text{(d.o. stabl.)} \left| \sum_{d\in d(\underline{\mathcal{T}})} f_{a,d} - \sum_{d\in d(\overline{\mathcal{T}})} f_{a,d} \right| \le 1 \quad \forall a\in\mathcal{A} \quad (20x)$$

$$y_{e,t} \in \{0,1\} \quad (20y)$$

$$v_{a,t}, f_{a,d}, b_{c,t} n_{c,d} o_{a,d} \in [0,1] \quad (20z)$$

$$h_{a,d}, \underline{h}_{a,d}\overline{h}_{a,d}, w_c, \text{def}_{t,\le w}, a_{t,w} \in \mathbb{R}^+ \quad (20aa)$$

Constraint (20c) constrains the auxiliary variable $v_{a,t}$ properly. Constraint (20d) treats entity conflicts in a slightly changed formulation, to also constrain variable $f_{a,d}$ properly. Constraints (20e)–(20g) define the lower bound on room allocation, and are similar to those previously described. Constraint (20h) ensures the assigning of events are locked to a certain timeslot. Constraints (20i) and (20j) ensure the placement of events which must be placed in the same/contiguous timeslots. Constraints (20k) and (20l) ensure that variable $n_{c,d}$ is constrained properly, and that no more than N_c neighbor-day conflicts are scheduled for class $c\in\mathcal{C}$. Constraint (20m) poses restrictions on timeslots for which an entity is unavailable. Constraints (20n)–(20p) ensure that idle slots for entities are penalized accordingly. Constraint (20q) ensures that sufficient days off are assigned to each entity. Constraint (20r) makes sure that if an entity $a\in\mathcal{A}$ has no event on some day $d\in\mathcal{D}$, then variable $f_{a,d}$ is forced to take value 1. This is necessary as this variable is minimized in the objective. Constraints (20s) and (20t) ensure that day-conflicts for classes do not occur, and constrains the variable $b_{c,t}$ properly. Constraint (20u) ensures that the limit on the amount of events assigned to a day for entity a is respected. For an entity, days with only one event scheduled are undesirable. Constraint (20v) penalizes days with only one events scheduled for entity a. Constraint (20w) forces week-stability for events of classes, i.e. in case two-weeks are being planned, events for courses must be spread evenly throughout the weeks. Constraint (20x) ensures that in case several weeks are being planned, the days off for an entity are spread evenly throughout the weeks.

5.2. Stage II

Let variable $v_{c,r}\in\{0,1\}$ take value 1 if there is at least one event of which class $c\in\mathcal{C}$ participates assigned to room $r\in\mathcal{R}$, and 0 otherwise. Variable $s_c\in\mathbb{N}_0$ is the amount of 'excess' rooms assigned to events of class $c\in\mathcal{C}$, i.e. the total amount of rooms assigned minus one. This is used to enforce room stability for classes, since it is undesirable for a class to be assigned too many different rooms. Parameter $LR_{e,r}\in\{0,1\}$ takes value 1 if event $e\in\mathcal{E}$ is locked room $r\in\mathcal{R}$.

Stage II Lectio (21)

Table 1
Computational results. For each dataset is shown the objective 'Obj' obtained by each method. For the IP-based approaches, also the best found lower bounds 'LB' are shown (i.e. for the 3-index model is shown the final value of the LP-relaxation used internally by Gurobi, and for TSD is shown the final value of the LP-relaxation of the Stage I model, as described in Section 4). If a solution is best overall, it is marked in **bold**. If a bound is best overall, it is marked with a '*'. For the two-stage approach of this paper is shown the runtime 'Time' and final gap 'Gap' found by Gurobi for both Stage I and Stage II. For Stage II, column 'Diff.' denotes the difference between the lower bound for room allocation of Stage I, and the actual matching obtained by Stage II (excluding room stability). Column '$\overline{\text{Gap}}$' denotes the best overall gap, i.e. the gap between the best available solution and the best available bound.

Dataset	Previous methods					$\text{TSD}^{\text{RoomLB}}$							
	ALNS	3-index model		TSD		Stage I		Stage II					
	Obj	Obj	LB	Obj	LB	Time	Gap	Time	Gap	Diff.	Obj	LB	$\overline{\text{Gap}}$
AalborTG2012	6317	6118	*5946	6018	5934	> 6480	0.7	4	0.0	0	**6005**	5941	1.0
AarhusA2011	**10 037**	58 015	–	15 872	*5986	> 6480	66.6	154	0.0	160	18 122	5985	40.4
AarhusA2012	**7971**	17 096	5722	8947	*6005	> 6480	49.4	108	0.0	48	11 936	5962	24.7
Aars2009	**14 900**	49 504	–	20 780	11 874	> 6480	47.9	14	0.0	0	24 240	*12 641	15.2
Aars2010	**16 268**	81 970	–	25 057	13 134	> 6480	42.7	22	0.0	1	24 692	*14 151	13.0
Aars2011	**14 256**	77 967	–	30 623	9709	> 6480	68.9	10	0.0	3	33 790	*10 501	26.3
Aars2012	**10 701**	55 049	–	21 206	7456	> 6480	60.3	21	0.0	1	20 274	*8044	24.8
Alssund2010	**9967**	52 717	–	23 173	6811	> 6480	67.9	324	0.0	8	21 455	*6876	31.0
Alssund2012	**29 803**	108 810	–	108 810	–	> 6480	–	6	0.0	0	108 810	–	–
BagsvaG2010	3960	6777	3171	**3916**	3063	> 6480	19.4	14	0.0	9	4051	*3227	17.6
BirkerG2011	**42 063**	119 600	–	119 600	–	> 6480	–	7	0.0	0	119 600	–	–
BirkerG2012	19 552	110 180	–	19 322	15 662	> 6480	0.9	> 720	1.0	16	**18 182**	*17 709	2.6
BjerrG2009	**16 877**	52 639	–	35 514	11 094	> 6480	55.2	11	0.0	0	27 396	*12 288	27.2
BjerrG2010	**4983**	12 868	*3928	5788	3868	> 6480	33.1	13	0.0	37	5977	3925	21.2
BjerrG2011	**6334**	13 009	*4142	9302	4060	> 6480	64.8	97	0.0	20	11 676	4079	34.6
BjerrG2012	**8023**	17 200	*5055	15 265	5007	> 6480	71.2	160	0.0	16	17 404	4991	37.0
BroendG2012	2040	2005	*1881	1929	1859	1028	0.0	17	0.0	5	**1928**	1877	2.4
CPHWGym2010	**6775**	34 415	–	19 363	*3759	> 6480	77.4	11	0.0	0	16 589	3752	44.5
CPHWGym2011	**5679**	38 232	–	16 212	4095	> 6480	72.7	10	0.0	0	15 046	*4103	27.8
CPHWGym2012	**6762**	40 945	–	15 543	4205	> 6480	75.5	19	0.0	1	17 194	*4215	37.7
CPHWHG2012	**11 077**	46 625	8157	23 088	*8338	> 6480	64.1	16	0.0	0	23 219	8326	24.7
CPHWHTX2010	**11 342**	27 174	9179	15 943	8828	> 6480	52.1	7	0.0	0	19 314	*9259	18.4
CPHWHTX2011	20 734	22 466	20 460	20 708	18 490	> 6480	0.6	6	0.0	11	**20 632**	*20 470	0.8
CPHWHTX2012	**16 256**	25 998	14 481	21 392	13 115	> 6480	35.4	4	0.0	0	22 481	*14 531	10.6
DetFG2012	7560	8017	*7168	7265	7018	> 6480	0.7	8	0.0	68	**7258**	7116	1.2
DetKG2010	**2947**	6058	1732	4006	*1821	> 6480	55.6	3	0.0	4	4102	1814	38.2
DetKG2011	**2820**	5594	1732	4366	1780	> 6480	60.9	2	0.0	2	4577	*1781	36.8
EUCN2009	**3737**	7557	2911	4298	2856	> 6480	40.3	4	0.0	0	5001	*2982	20.2
EUCN2010	3882	4231	3329	3463	3246	> 6480	1.4	6	0.0	1	**3430**	*3375	1.6
EUCN2011	1468	1435	*1395	1430	1384	> 6480	2.2	1	0.0	2	**1426**	1384	2.2
EUCN2012	**3289**	9430	2327	5059	*2363	> 6480	60.2	4	0.0	0	5913	2359	28.2
EUCNHG2010	1505	1476	1371	1421	1368	> 6480	2.1	2	0.0	0	**1408**	*1378	2.1
EUCS2012	3714	4689	3576	3783	3347	> 6480	3.0	3	0.0	0	**3695**	*3584	3.0
FaaborgG2008	**68 124**	125 330	–	125 330	–	> 6480	–	14	0.0	0	125 330	–	–
FalkonG2009	**10 449**	88 890	–	88 890	–	> 6480	–	5	0.0	0	88 890	–	–
FalkonG2011	**8584**	76 170	–	16 543	*5183	> 6480	75.9	> 720	0.0	48	20 758	4953	39.6
FalkonG2012	**10 143**	100 190	–	16 666	*6105	> 6480	58.6	> 720	0.1	121	14 908	6050	39.8
GUAasia2010	6527	6579	6354	6461	6035	26	0.0	> 720	0.1	5	**6422**	*6374	0.7
GUQaqor2011	**6674**	19 623	4537	10 005	*4554	> 6480	59.9	3	0.0	18	11 396	4542	31.8
GUQaqor2012	**5733**	11 488	4314	7619	4294	> 6480	55.1	10	0.0	0	9650	*4324	24.6
HadersK2011	**7128**	51 190	–	14 229	*3909	> 6480	76.2	> 720	0.0	43	16 494	3888	45.2
HasserG2010	**11 963**	96 790	–	96 790	–	> 6480	–	6	0.0	0	96 790	–	–
HasserG2011	**16 061**	99 840	–	99 840	–	> 6480	–	6	0.0	0	99 840	–	–
HasserG2012[a]	**18 338**	112 160	–	112 034	–	> 6480	–	7	0.0	0	112 160	–	–
HerningG2010	**37**	**37**	*37	**37**	35	0	0.0	1	0.0	0	**37**	35	0.0
HerningG2011	**15 091**	163 785	–	23 117	*9829	> 6480	61.8	108	0.0	169	26 410	9746	34.9
HerningG2012	**13 147**	185 433	–	14 952	9763	> 6480	48.4	> 720	0.1	262	19 834	*9817	25.3
HoejeTaG2008	2958	6292	2253	**2707**	2563	> 6480	6.4	3	0.0	0	2775	*2587	4.4
HoejeTaG2009	**9157**	45 260	–	26 066	*5773	> 6480	79.7	105	0.0	3	27 779	5628	37.0
HoejeTaG2010	**9862**	45 095	–	25 678	*6188	> 6480	78.1	106	0.0	5	27 886	6116	37.3
HoejeTaG2011	**10 158**	51 050	–	32 630	*6726	> 6480	78.2	66	0.0	3	30 327	6601	33.8

Table 1 (*continued*)

Dataset	Previous methods					TSD^{RoomLB}							
	ALNS	3-index model		TSD		Stage I		Stage II					
	Obj	Obj	LB	Obj	LB	Time	Gap	Time	Gap	Diff.	Obj	LB	$\overline{Gap}$
HoejeTaG2012	**12 502**	72 455	7592	18 627	7845	> 6480	79.8	9	0.0	3	39 326	*7952	36.4
HorsenS2009	3111	**3100**	*3100	**3100**	2865	1	0.0	4	0.0	13	**3100**	3059	0.0
HorsenS2012	**10 056**	86 090	–	86 090	–	> 6480	–	3	0.0	0	86 090	–	–
Johann2012	**23 001**	92 575	–	27 781	18 456	> 6480	33.5	233	0.0	6	29 491	*19 590	14.8
KalundG2011	**38479**	126 150	–	126 150	–	> 6480	–	9	0.0	0	126 150	–	–
KalundG2012	**26 768**	123 010	–	123 010	–	> 6480	–	11	0.0	0	123 010	–	–
KalundHG2010	**5631**	12 103	4540	6351	4551	> 6480	29.7	6	0.0	0	6605	*4642	17.6
KoebenPG2012	888	1872	637	**874**	642	> 6480	37.9	1	0.0	2	1052	*645	26.2
KoegeH2012	**11 418**	108 347	–	20 150	9096	> 6480	53.7	12	0.0	0	20 390	*9440	17.3
KongshoG2010	**4296**	8889	2411	7954	*2488	> 6480	65.9	30	0.0	0	7208	2451	42.1
MariageG2009	**8013**	54 030	–	20 138	5118	> 6480	69.7	> 720	0.0	18	17 506	*5286	34.0
MorsoeG2012	**5651**	42 762	–	10 241	3854	> 6480	66.0	23	0.0	6	11 674	*3947	30.2
NaerumG2008	**24 104**	118 370	–	117 894	–	> 6480	–	7	0.0	0	117 894	–	–
NaerumG2009	7667	100 450	–	6681	*5114	> 6480	0.3	> 720	6.2	0	**5466**	5113	6.4
NielsSG2011	**4953**	10 464	3323	6132	*3412	> 6480	37.6	9	0.0	0	5397	3367	31.1
NielsSG2012	**6952**	12 747	5722	8003	*5738	> 6480	37.6	14	0.0	4	9192	5724	17.5
NordfynG2012	5160	8201	*4152	**4890**	4048	> 6480	23.3	35	0.0	59	5510	4107	15.1
NyborgG2011	**13 944**	94 059	–	31 809	*6129	> 6480	–	7	0.0	4	85 816	–	56.0
OdderCfU2010	**18 219**	59 540	–	40 032	12 188	> 6480	66.9	66	0.0	2	38 875	*12 865	29.4
OdderG2009	**9308**	59 851	–	57 586	–	> 6480	78.1	67	0.0	67	24686	*5361	42.4
OdderG2012	**12 307**	17 402	9602	14 888	8878	> 6480	64.2	4	0.0	57	27 199	*9688	21.3
OrdrupG2010	13 663	75 700	–	**12 936**	10 665	> 6480	39.5	10	0.0	0	18 101	*10 810	16.4
OrdrupG2011	**21 612**	1 16 400	–	31 329	16 904	> 6480	38.7	305	0.0	8	28 884	*17 692	18.1
RibeK2011	**21 679**	61 945	–	43 175	16 209	> 6480	53.8	229	0.0	5	39 107	*18 055	16.7
RysenG2010	**39 971**	110 690	–	110 690	–	> 6480	–	6	0.0	0	110 690	–	–
RysenG2011	**22 260**	100 313	–	25 989	17 756	> 6480	71.4	9	0.0	5	68 927	*19 725	11.4
RysenG2012	**19 841**	110 111	–	22 156	15 115	> 6480	71.7	14	0.0	15	59 124	*16 708	15.8
SanktAG2012	4207	4624	3415	3911	3376	> 6480	0.7	> 720	0.5	39	**3721**	*3538	4.9
SkanderG2010	7209	7708	6051	6875	5712	> 6480	0.6	> 720	0.5	72	**6485**	*6238	3.8
SkanderG2011	**22 525**	88 470	–	88 470	–	> 6480	–	5	0.0	0	88 470	–	–
SkanderG2012	**20 138**	98 487	–	95 319	–	> 6480	–	7	0.0	3	95 319	–	–
SkiveG2010	**43 120**	194 740	–	194 740	–	> 6480	–	526	0.0	0	194 740	–	–
SlagelG2012[a]	**32 167**	162 960	–	162 765	–	> 6480	–	417	0.0	0	162 960	–	–
SoendS2011	**11 776**	83 560	–	83 560	–	> 6480	–	131	0.0	0	83 560	–	–
SoendS2012	**8420**	17 778	*6838	11 915	6647	> 6480	72.5	8	0.0	4	24 668	6739	18.8
StruerS2012	**73 361**	–	–	207 488	–	> 6480	–	700	0.0	0	211 960	–	–
VardeG2012	**10 777**	20 933	*5921	20 622	5720	> 6480	72.1	12	0.0	2	20 496	5668	45.1
VejenG2009	**11 264**	69 450	–	69 450	–	> 6480	73.9	> 720	0.0	7	27 954	*7290	35.3
Vejlefjo2011	**13 514**	52 035	–	18 043	8511	> 6480	60.0	456	0.0	3	22 066	*8805	34.8
VestfynG2009	5973	11 606	4176	5999	4137	> 6480	14.5	553	0.0	18	**5032**	*4211	16.3
VestfynG2010	6761	16 895	*4308	5974	4225	> 6480	16.3	65	0.0	21	**5239**	4290	17.8
VestfynG2011	7013	13 624	5110	6657	4925	> 6480	19.6	38	0.0	24	**6522**	*5159	20.9
VestfynG2012	5244	11 095	4279	**5212**	4210	> 6480	17.5	149	0.0	21	5319	*4315	17.2
ViborgK2011	**14 923**	99 170	–	99 170	–	> 6480	–	6	0.0	0	99 170	–	–
ViborgTG2009	**10 216**	19 891	8695	12 077	8356	> 6480	34.7	45	0.0	3	13 387	*8740	14.4
ViborgTG2010	**4932**	12 727	4130	10 226	3990	> 6480	60.9	11	0.0	19	10 665	*4146	15.9
ViborgTG2011	**7478**	16 433	6716	9808	6204	> 6480	38.8	12	0.0	13	11 088	*6772	9.4
VirumG2012	**27 738**	140 883	–	32 183	17 770	> 6480	75.3	16	0.0	9	79 111	*19 486	29.7
VordingbG2009	**8568**	17 025	5457	9905	5243	> 6480	33.6	10	0.0	167	8972	*5787	32.5
Avg.							44.3		0.1	17.9			22.3

[a] An artificial bound on the amount of Halls' inequalities generated was enforced for tractability, see Section 4.3.

$$\min w^{II} = \sum_{e \in \mathcal{E},\ r \in \mathcal{R}} \pi_{e,r} z_{e,r} + \varepsilon \sum_{c} s_c \tag{21a}$$

s.t.

$$\text{(one room)} \sum_{r \in \mathcal{R}} z_{e,r} = 1 \quad \forall e \in \mathcal{E} \tag{21b}$$

$$\text{(room conf.)} \sum_{e \in \mathcal{E}} y^*_{e,t} z_{e,r} \le G_{r,t} \quad \forall r \in \mathcal{R} \backslash r_D, t \in \mathcal{T} \backslash t_D \tag{21c}$$

$$\text{(eligible rooms)}\ z_{e,r} \le K_{e,r} \quad \forall e \in \mathcal{E}, r \in \mathcal{R} \tag{21d}$$

$$\text{(locked rooms)}\ z_{e,r} = 1 \quad \forall e \in \mathcal{E}, r \in \mathcal{R}, LR_{e,r} = 1 \tag{21e}$$

$$\text{(room stbl.)} \sum_{e \in \mathcal{E},\ t \in \mathcal{T} \backslash t_D} J_{e,c} y^*_{e,t} z_{e,r} - \sum_{e \in \mathcal{E}} J_{e,c} v_{c,r} \le 0 \quad \forall r \in \mathcal{R} \backslash r_D, c \in \mathcal{C} \tag{21f}$$

$$\text{(room stbl.)} \sum_{r \in \mathcal{R}} v_{c,r} - 1 \le s_c \quad \forall c \in \mathcal{C} \tag{21g}$$

$$\text{(not only room)} \sum_{r \in \mathcal{R} \backslash r_D} y^*_{e,t_D} z_{e,r} - \sum_{r \in \mathcal{R}} LR_{e,r} \le 0 \quad \forall e \in \mathcal{E} \tag{21h}$$

$$z_{e,r}, v_{c,r} \in \{0,1\} \tag{21i}$$

$$s_c \in \mathbb{R}^+ \tag{21j}$$

Constraint (21e) ensures that events with locked rooms are assigned accordingly. Constraints (21f) and (21g) constrains variables $v_{c,r}$ and s_c properly, and thereby penalizes room stability. Constraint (21h) enforces that an event cannot be assigned a room if it not assigned to a timeslot, unless the event is locked to a specific room.

Notice that the room stability constraints (21f) and (21g) of Model (21) are not handled in any way in the Stage I model. Additional constraints which handle these constraints would theoretically improve the decomposition. It is however not trivial to model these constraints as a matching problem in a bipartite graph, so another approach may be required. This is a subject for future research. Apart from the room stability constraints, all other constraints are optimally handled by the decomposition, with the exception of the room allocation penalties, which are only partially integrated in the Stage I model.

6. Computational results

For implementation purposes, Gurobi 5.0.1 was used as MIP solver on a machine with an Intel Core i7 930@2.80 GHz CPU and 12 GB of RAM, running Windows 8 64bit. Default parameter settings were used, and the interface was C# 4.5. The problem instances have been taken directly from the Lectio database, and are the same ones used in Sørensen and Stidsen [31]. These 100 real-world datasets provide a substantial ground for concluding on the numerical experiments. Note that 3 of these instances are available in the XHSTT format [25] at http://www.utwente.nl/ctit/hstt/. We plan to make additional datasets available in this format in the future.

A time limit of 7200 s was imposed (6480 s for Stage I, and 720 s for Stage II), and Gurobi was allowed to use 8 threads. For the Stage I model, the initial solution given to Gurobi consists of assigning all events to the dummy-timeslot, except for those events locked to a specific timeslot. The initial solution for the Stage II model is analogous; Events are assigned to the dummy-room or the room which the event is locked too. A single run was used to establish results, as Gurobi has deterministic behavior.

Two other solution approaches are described for the same timetabling problem in Sørensen and Stidsen [31]. These are used in comparison with the algorithm of this paper, and are briefly described below:

- The 'usual' formulation using a three-index binary variable, denoted 3-*index model* in the following. This is solved using Gurobi with standard settings, with a time limit of 7200 s. The objectives listed are the result of a single run.
- An *Adaptive Large Neighborhood Search* heuristic, denoted *ALNS* in the following. The reported objectives for this method is the average obtained over 10 runs, each run with a timelimit of 240 s. Hence, the comparison of objectives w.r.t. this heuristic is not 'fair', but it will be seen that even with this shorter timelimit, the ALNS in general performs best. This method is the one currently used by the customers of Lectio.

Furthermore, we test the described decomposition both with and without the room penalties added to the Stage 1 model (i.e. Model (20) with and without Eqs. (20e), (20f) and (20g)). Thereby an empirical test of the effect of extending Stage I is performed. In the following, these two methods are denoted *TSD* and *TSD*$^{\text{RoomLB}}$, respectively. Notice that the results for TSD can also be found in the technical report [31].

Table 1 shows the obtained results. Table 2 summarizes some key numbers. Table 3 gives a summary for the three exact methods. A gap between an IP objective z and a lower bound LB is calculated by $100(z-LB)/z$.

A number of conclusions can be drawn from the numbers:

- For 97 instances, the solution obtained by TSD$^{\text{RoomLB}}$ is at least as good as the solution obtained by the 3-index model, and for most instances significantly better.
- TSD$^{\text{RoomLB}}$ is generally the best method for generating bounds, finding the best bound on 49 instances overall.
- TSD$^{\text{RoomLB}}$ was capable of finding a lower bound for 80 instances. This means that for 20 instances, Gurobi was unable to solve the LP-relaxation of the root node of the Stage I model within the timelimit. Table 4 shows statistics for these instances (the presolved models). It is seen that the problems do not contain coefficients of huge magnitude in neither the objective, system matrix, or rhs. Hence the issue seems related to the relatively big number of constraints and variables. In average, these instances have more than 100 000 constraints and variables, hence we think its fair to consider them as large-scale. Note that if the root-LP was not solved for an instance, the reported solution replicates the initial solution provided by us to Gurobi (Gurobi apparently starts its solution process by verifying the feasibility of the MIP Start attributes).

Table 2
Results summary. Note that rows 'Best solution' and 'Best bound' also counts draws.

	ALNS	3-index model	TSD	TSD$^{\text{RoomLB}}$
Solution found	100	99	100	100
Best solution	77	2	8	18
Bound found	–	46	79	80
Best bound	–	13	19	49

Table 3
Comparison of the amount of best found solutions for the exact methods. Draws are also counted.

	3-index model	TSD	TSD$^{\text{RoomLB}}$
Best solution	16	66	52

Table 4
Statistics of the Stage I models (after presolve) where Gurobi was unable to solve the LP-relaxation of the root-node within the timelimit (i.e. for those instances where the TSD^{RoomLB}) was unable to provide a lower bound). Column 'Cons.' shows the number of constraints and 'Non-zeros' shows the amount of non-zeros in the model. 'Variables' shows the amount of continuous, integer and binary variables. 'Obj. coef.', 'Model coef.', and 'RHS coef' shows the smallest and largest coefficient in the objective function, system matrix and right-hand side, respectively.

	Cons.	Non-zeros	Variables			Obj. coef.		Model coef.		RHS coef.	
			Cont.	Integer	Binary	Min.	Max.	Min.	Max.	Min.	Max.
Min.	73 293	881 433	24 089	52 518	41 121	1	84	1	4	0	6
Max.	167 978	4 200 206	52 015	252 514	246 874	1	1120	1	16	1	103
Avg.	118 396	1 967 455	36 918	101 460	88 321	1	264	1	7	1	32

- TSD^{RoomLB} produces the best solution for 18 instances overall, while TSD produces the best solution on 8 instances. The ALNS heuristic is best on 77 instances, and is currently the best algorithm for this problem (keep in mind the ALNS algorithm was allowed significantly less CPU time).
- Comparing the exact methods (Table 3), it is seen that it is generally not profitable to use the extended Stage I model if the goal is to obtain good solutions. Both variants of the decomposition finds more best solutions than the pure 3-index model.
- The Stage I model of TSD^{RoomLB} is a challenge for Gurobi, with an average gap of 44.3% over all instances where a LB was found. Only 4 instances are solved to optimality, and these are among the smallest instances (see [31] for instance statistics).
- The Stage II model of TSD^{RoomLB} is in general easy to solve. The average gap for this model over all problem instances is 0.1%, and 89 instances are solved to optimality. This means that future research can focus on solving the Stage I model.
- The difference between the lower bound on room allocation and the actual allocation of rooms (column 'Diff.') is low, compared to the magnitude of objectives in general. This means that only a small increase in solution quality can be gained by improving the bound on room allocation.

As an extension to the decomposition, one could use the ALNS heuristic to provide a starting solution. Since the ALNS heuristic is able to produce a fairly good solution quickly, this would most likely lead to improved performance.

As a loose remark, we mention that Burke et al. [7] formulate an IP of the *Udine Course Timetabling Problem* (used in the International Timetabling Competition 2007), using a three-indexed binary variable, and reports that CPLEX 11 uses up to 6400 s when solving the root LP (using Dual Simplex, which is also used by Gurobi as default). Their model is quite similar in structure to ours, so possibly this class of IP formulations contain undesirable properties in the eyes of general-purpose MIP solvers.

7. Conclusion

A Two-Stage Decomposition for a real-world high school timetabling problem has been shown. This splits the Integer Programming model into two smaller models, which reduces the number of variables significantly. Computational results show that this approach is way more effective than solving the usual original IP with a 3-index binary variable, in terms of both the obtained solutions and the obtained bounds. This constitutes the TSD as the best exact method for solving this particular timetabling problem. However, the integration of the lower bound on room allocation in the Stage I model has bad influence on the quality of solutions, but makes the decomposition capable of achieving better lower bounds. Nevertheless this extension of the Stage I model represents interesting theory which can likely be used in the context of decomposing other timetabling problems.

For other types of (timetabling) problems, this type of decomposition might be a way of enhancing computational times. However, a special structure is required for applying the decomposition, which limits the set of applicable problems. On the other hand, the advantage gained by reducing the number of variables should not be underestimated, and we encourage researchers to attempt this type of decomposition if possible.

Approximating the room allocation penalties in the Stage I model is an interesting approach, and also sets a possible agenda for future work; (1) Can a better approximation (or even the exact value) be found for the minimum weight maximum matching problem representing room penalties? (2) Can the room stability penalties be incorporated in the Stage I model? However, the most important issue for future research is a more efficient way of solving the Stage I model. This is the bottleneck of the TSD for this particular problem.

References

[1] Avella P, DAuria B, Salerno S, Vasilâev I. A computational study of local search algorithms for Italian high-school timetabling. Journal of Heuristics 2007;13:543–56.
[2] Badri MA. A two-stage multiobjective scheduling model for [faculty-course-time] assignments. European Journal of Operational Research 1996;94(1):16–28.
[3] Balas E, Pulleyblank W. The perfectly matchable subgraph polytope of a bipartite graph. Networks 1983;13(4):495–516.
[4] Bardadym V. Computer-aided school and university timetabling: the new wave. In: Burke E, Ross P, editors. Practice and theory of automated timetabling, lecture notes in computer science, vol. 1153. Berlin, Heidelberg: Springer; 1996. p. 22–45.
[5] Birbas T, Daskalaki S, Housos E. Timetabling for greek high schools. Journal of the Operational Research Society 1997;48 1191–1200(10).
[6] Birbas T, Daskalaki S, Housos E. School timetabling for quality student and teacher schedules. Journal of Scheduling 2009;12(April):177–97.
[7] Burke E, Marecek J, Parkes A, Rudová H. Decomposition, reformulation, and diving in university course timetabling. Computers & Operations Research 2010;37(3):582–97.
[8] Burke E, Newall JP. A multistage evolutionary algorithm for the timetable problem. IEEE Transactions on Evolutionary Computation 1999;3(1):63–74.
[9] Carter M. A decomposition algorithm for practical timetabling problems. Technical report. 83-06, Department of Industrial Engineering, University of Toronto; 1983.
[10] Daskalaki S, Birbas T. Efficient solutions for a university timetabling problem through integer programming. European Journal of Operational Research 2005;160(1):106–20.
[11] Daskalaki S, Birbas T, Housos E. An integer programming formulation for a case study in university timetabling. European Journal of Operational Research 2004;153:117–35.
[12] de Werra D. An introduction to timetabling. European Journal of Operational Research 1985;19(2):151–62.
[13] Dimopoulou M, Miliotis P. Implementation of a university course and examination timetabling system. European Journal of Operational Research 2001;130(1):202–13.
[14] Edmonds J. Maximum matching and a polyhedron with 0, 1-vertices. Journal of Research of the National Bureau of Standards B 1965;69:125–30.
[15] Egerváry E. Matrixok kombinatorius tulajdonságairól. Matematikai és Fizikai Lapok 1931;38:16–28.
[16] Gotlieb CC. The construction of class-teacher timetables. In: Popplewell CM, editor. IFIP congress, vol. 62. North-Holland Publication Co.; 1962. p. 73–7.
[17] Lach G, Lübbecke M. Optimal university course timetables and the partial transversal polytope. In: McGeoch C, editor. Experimental algorithms, lecture

notes in computer science, vol. 5038. Berlin, Heidelberg: Springer; 2008. p. 235–48.
[18] Lach G, Lübbecke M. Curriculum based course timetabling: new solutions to udine benchmark instances. Annals of Operations Research 2012;194:255–72.
[19] Lawrie NL. An integer linear programming model of a school timetabling problem. The Computer Journal 1969;12(4):307–16.
[20] Lovász L, Plummer MD. Matching theory. AMS Chelsea Publishing; 2009.
[21] MirHassani S. A computational approach to enhancing course timetabling with integer programming. Applied Mathematics and Computation 2006;175 (1):814–22.
[22] Mittelman H. Benchmarks for optimization software. ⟨http://plato.asu.edu/bench.html⟩; August 2013. [Retrieved 20/8-2013].
[23] Papoutsis K, Valouxis C, Housos E. A column generation approach for the timetabling problem of greek high schools. The Journal of the Operational Research Society 2003;54(3):230–8.
[24] Pillay N. A survey of school timetabling research. Annals of Operations Research 2013(February):.
[25] Post G, Ahmadi S, Daskalaki S, Kingston J, Kyngas J, Nurmi C, et al. An xml format for benchmarks in high school timetabling. Annals of Operations Research 2012;194:385–97.
[26] Post G, Gaspero LD, Kingston JH, McCollum B, Schaerf A. The third international timetabling competition. In: Proceedings of the ninth international conference on the practice and theory of automated timetabling (PATAT 2012). Son, Norway; August 2012.
[27] Qualizza A, Serafini P. A column generation scheme for faculty timetabling. In: Burke E, Trick M, editors. Practice and theory of automated timetabling V, lecture notes in computer science, vol. 3616. Berlin, Heidelberg: Springer; 2005. p. 161–73.
[28] Santos H, Uchoa E, Ochi L, Maculan N. Strong bounds with cut and column generation for class-teacher timetabling. Annals of Operations Research 2012;194(April (1)):399–412.
[29] Schaerf A. A survey of automated timetabling. Artificial Intelligence Review 1999;13:87–127.
[30] Schrijver A. Combinatorial optimizationpolyhedra and efficiency, algorithms and combinatorics. Springer; 2003.
[31] Sørensen M, Stidsen T. Comparing solution approaches for a complete model of high school timetabling. Technical report. 5.2013, DTU Management Engineering. Technical University of Denmark; March 2013.