An Integer Programming Formulation for the Music School Timetabling Problem.

E. I. Ásgeirsson · Þ. Gunnarsdóttir

Abstract Research on educational timetabling has focused mostly on high-school timetabling, university timetabling and examination timetabling. We introduce a new problem, the music school timetabling problem which is a hybrid between the high-school and university timetabling problems, sharing many features from both problems. The music school timetabling problem contains both group courses and private lessons that must be scheduled, with limitations on the availability of both teachers and students. We give an integer programming formulation for the music school timetabling problem and present the results from a case study from Tónmenntaskóli Reykjavíkur, a music school in Reykjavík, Iceland. The results show that, although the problem is challenging, we can still get high quality timetables using a two-phased approach, where we focus first on group courses at the music school, before scheduling private lessons.

Keywords Educational Timetabling · Music School · Integer Programming

1 Introduction

The problem of creating timetables for educational institutions and other organizations is a well known scheduling problem and has been widely studied since the 1960s [2]. Burke et. al. gave the following definition of the general timetabling problem:

"A timetabling problem is a problem with four parameters: T, a finite set of times; R, a finite set of resources; M, a finite set of meetings;

E. I. Ásgeirsson
School of Science and Engineering & ICE-TCS, Reykjavik University
E-mail: eyjo@ru.is

Þ. Gunnarsdóttir
School of Science and Engineering, Reykjavik University
E-mail: thorhildurg12@ru.is

and C, a finite set of constraints. The problem is to assign times and resources to the meetings so as to satisfy the constraints as far as possible." [3]

For the educational timetabling problem, the meetings represent the courses while the resources include both teachers, students, classrooms and perhaps some special equipment.

Educational timetabling [7] is often split into high-school timetabling [11] (or just school-timetabling) and university timetabling [9]. The main difference between high-school and university timetabling is that for the university timetabling, each student follows an individual timetable, while in the high-school timetabling problem the students are grouped together into student groups where all students in the group have the same program [6].

Music schools face the same scheduling problems as other schools and educational institutions, that is, each semester they need to create a timetable that specifies the schedule for all courses, students and teachers at the school, determining where and when each course should be taught. The music school timetabling problem shares many aspects of the school-timetabling problem and the university timetabling problem while other features of the problem are unlike both the high-school and university timetabling problems.

The music school timetabling problem has not been studied widely in the research community, while there is extensive literature on both the high-school timetabling problem [11] and the university timetabling problem [5,1,13,8].

In this paper, we will introduce the music school timetabling problem (Section 2) and look at a two-phase integer programming formulation that gives a good solution for the problem (Section 3), as shown by a case-study using real data from a music school in Reykjavik, Iceland (Section 4).

2 The Music School Timetabling Problem

Each semester music schools must create a timetable that specifies when and where each course should take place, which students should attend which course and who the teacher is. The curriculum at a music school usually consists of both group courses, such as music theory, and private lessons where the student is alone with the teacher. Additionally, there are often band sessions for some of the students. Unlike the high-school timetabling problem, the music school timetabling problem only have a limited group of students that have the same program. Also, the music school timetabling problem is unlike the university timetabling problem as we cannot allow conflicts between courses, each student must be able to attend all the courses in their program. Students in music schools follow different programs based on their age, proficiency and their choice of a musical instrument. The youngest students usually take only group lessons, often flute preschool, while older students that have selected their musical instrument take both music theory in groups as well as private lessons. Traditionally, students keep the same private teacher throughout their studies, so private teachers are assigned apriori for most of the students.

For the educational timetabling problem, one of the assuptions that is usually made is that all the students are available at any time during school hours, while the teachers are sometimes allowed to have a more limited availability. Since the music school lessons usually take place after the regular school is over, i.e. in the afternoon or even during the evenings, the music schools often cannot assume that the students will be available at any specific time. Additionally, the music schools are often competing with other extra-curricular activities, such as sports or art lessons. Therefore, both students and teachers can have limitations on their availability during school hours.

The music school timetabling problem has both hard and soft constraints, although the majority of the constraints for the music school timetable fall under the category of hard constraints. The objective is to find a feasible solution that satisfies all of the hard constraints and minimizes the violations of the soft constraints and other penalties on the solution.

2.1 Hard Constraints

To be considered a feasible timetable for a music school, the schedule must satisfy the following hard constraints:

- **Required courses**: Each student must be assigned to all their required courses (both group and private courses).
- **Student conflicts**: A student cannot have any conflicts between courses in their schedule.
- **Teacher conflicts**: Teachers cannot teach two courses at the same time, so there cannot be any conflicts in the teachers' schedules.
- **Resources**: Each classroom must be of sufficient size for the course and have necessary equipment (e.g. grand piano).
- **Student availability**: Students can have limited availability; a course cannot be scheduled unless the students are available.
- **Teacher availability**: Teachers can also have limited availability so a course cannot be scheduled unless the teacher is available.
- **Practice days**: Private sessions should not be scheduled on consecutive days; the students must have time to practice before next private session.

2.2 Soft Constraints and Objective Function

The soft constraints can vary from one music school to another. These conditions can either be formulated as regular constraints with penalties for any violations, or directly included into the objective function. Common soft constraints are:

- **Adjacent sessions**: A private sessions for a student should be adjacent or close to adjacent to the group course for that student when possible.

– **Young priority**: Younger students should be scheduled earlier in the day than older students.

The courses at a typical music school vary in the number of consecutive timeslots that they need. A private session usually takes a single timeslot, while music theory and group courses usually take two timeslots. Some group courses are also taught more than once during the week and any students assigned to those courses must be assigned to all of these sessions.

3 The Music School Timetabling Problem

To formulate the music school timetabling problem in full detail, we would need binary variables that specify whether student s is assigned to course c with teacher k in classroom r in timeperiod t on day d. However, using this detailed formulation is both time consuming and unneccesarily complicated for most cases. In reality, most of the teacher-student connections have already been decided, and in many cases it is also sufficient to ensure that the number of concurrent courses that need a specific type of classroom stays under some limit.

We fix beforehand which teachers should handle each course, and instead of assinging courses directly to classrooms, we make sure that for each course category (e.g. piano lessons or group music theory) the number of concurrenly scheduled courses does not exceed the number of classrooms that can be used for those courses. Deciding beforehand the teacher for each course and private lesson is helpful since it simplifies the problem, but it is also something that most music schools would like to decide manually, instead of having a computer program specify which teacher is teaching which course and which student.

3.1 IP Model for the Music School Timetabling Problem

We formulate the music school timetabling problem using a linear integer programming model. The main sets that we use are S, the set of students and C, the set of courses. In addition to these main sets, we also have the following parameters, sets and subsets:

– D: Number of days in the week, D is usually equal to 5.
– T: Number of timeslots per day, a typical timeslot is 30 or 45 minutes.
– $\alpha_{c,d,t}$: The penalty of assigning timeslot t on day d to student s. This parameter is useful in many ways, including giving priority to younger students to be scheduled earlier in the day.
– m_c: The maximum number of students that can be registered to course c.
– K_s^h, k_s^h: Student s must be assigned k_s^h times to the courses in $K_s^h \subseteq C$. Here $h = 1, \ldots, h_s$ represent the h_s different course types student s is assigned to. For example, the school might offer five music theory groups and three band sessions, and student s should be assigned to two of the music theory courses, two private lessons and one band session.

- OC_p: The set $OC_p \subseteq C, p = 1, \ldots, p_{\max}$ represents a set of courses that overlap, i.e. at most one course from OC_p can be scheduled at any time for $p = 1, \ldots, p_{\max}$. The set OC_p is useful to ensure that a teacher is not assigned to multiple courses at the same time.
- SA_s: Set of all timeslots where student s is not available, i.e. if $(d, t) \in SA_s$ then student s cannot be assigned to timeslot t on day d.
- CA_c: Course availability, if $(t, d) \in CA_c$ then course c cannot be scheduled in timeslot t on day d. Since teachers have already been asssigned to courses, the course availability reflects the availability of the assigned teacher.
- $DC \subseteq (C \times C)$: Double sessions, if $(c_i, c_j) \in DC$ then if course c_i is scheduled in timeslot t on day d, then course c_j should be scheduled in timeslot $t + 1$ on day d.
- $PC \subseteq (C \times C)$: Course pairs, if $(c_i, c_j) \in PC$ then any student assigned to course c_i should also be assigned to course c_j.
- CG_g, cg_g: The constant cg_g specifies how many courses in course group $CG_g \subseteq C$ can be scheduled simultaniously. The total number of course groups defined by the school is $g_{\max}$ and $g = 1, \ldots, g_{\max}$. For example, a course group can consist of all courses that require a classroom containing a grand piano, or all courses that have a large number of students and thus require a large classroom.
- $ND \subseteq (C \times C)$: If a pair of courses $(c_i, c_j) \in ND$ then the courses c_i and c_j must be scheduled on nonconsecutive days, i.e. there must be at least one day between those courses.

We use a main variable that can represent all possible solutions, as well as subsets of that variable that make the integer program easier to formulate. The main variable is

$$x_{s,c,d,t} = \begin{cases} 1 & \text{if student } s \text{ is assigned to course } c \text{ in timeperiod } t \text{ on day } d \\ 0 & \text{otherwise} \end{cases}$$

and the other variables are all subsets of the variable $x_{s,c,d,t}$:

$$y_{s,d,t} = \begin{cases} 1 & \text{if student } s \text{ is assigned to some course in timeperiod } t \text{ on day } d \\ 0 & \text{otherwise} \end{cases}$$

$$v_{c,d,t} = \begin{cases} 1 & \text{if course } c \text{ is scheduled in timeperiod } t \text{ on day } d \\ 0 & \text{otherwise} \end{cases}$$

$$w_{w,c} = \begin{cases} 1 & \text{if student } s \text{ is assigned to course } c \\ 0 & \text{otherwise} \end{cases}$$

The integer programming formulation for the music school timetabling problem is then:

$$\min \sum_{s \in S} \sum_{d=1}^{D} \sum_{t=1}^{T} \alpha_{s,d,t} y_{s,d,t} \tag{1}$$

$$\text{s.t.} \quad y_{s,d,t} \geq x_{s,c,d,t} \quad \forall s \in S, c \in C, d = 1, \ldots, D, t = 1, \ldots, T. \tag{2}$$

$$\sum_{c \in C} x_{s,c,d,t} \geq y_{s,d,t} \;\; \forall s \in S, d = 1,\ldots,D, t = 1,\ldots,T. \tag{3}$$

$$v_{c,d,t} \geq x_{s,c,d,t} \;\; \forall s \in S, c \in C, d = 1,\ldots,D, t = 1,\ldots,T. \tag{4}$$

$$\sum_{s \in S} x_{s,c,d,t} \geq v_{c,d,t} \;\; \forall c \in C, d = 1,\ldots,D, t = 1,\ldots,T. \tag{5}$$

$$w_{s,c} \geq x_{s,c,d,t} \;\; \forall s \in S, c \in C, d = 1,\ldots,D, t = 1,\ldots,T. \tag{6}$$

$$\sum_{d=1}^{D} \sum_{t=1}^{T} x_{s,c,d,t} \geq w_{s,c} \;\; \forall s \in S, c \in C. \tag{7}$$

$$\sum_{c \in C} x_{s,c,d,t} \leq 1 \;\; \forall s \in S, d = 1,\ldots,D, t = 1,\ldots,T. \tag{8}$$

$$\sum_{c \in OC_p} v_{c,d,t} \leq 1 \;\; \forall d = 1,\ldots,D, t = 1,\ldots,T, p = 1,\ldots,p_{\max}. \tag{9}$$

$$\sum_{s \in S} w_{s,c} \leq m_c \;\; \forall c \in C \tag{10}$$

$$\sum_{c \in K_s^h} w_{s,c} = k_s^h \;\; \forall s \in S, h = 1,\ldots,h_s. \tag{11}$$

$$v_{c_i,d,t} = v_{c_j,d,t+1} \;\; \forall (c_i,c_j) \in DC, d = 1,\ldots,D, t = 1,\ldots,T. \tag{12}$$

$$w_{s,c_i} = w_{s,c_j} \;\; \forall s \in S, (c_i,c_j) \in PC. \tag{13}$$

$$\sum_{c \in CG_g} v_{c,d,t} \leq cg_g \;\; \forall d = 1,\ldots,D, t = 1,\ldots,T, g = 1,\ldots,g_{\max}. \tag{14}$$

$$\sum_{t=1}^{T} \sum_{\hat{d}=d}^{d+1} v_{c_i,\hat{d},t} \leq 1 \;\; \forall d = 1,\ldots,D-1, \; (c_i,c_j) \in ND. \tag{15}$$

$$y_{s,d,t} = 0 \;\; \forall s \in S, (d,t) \in SA_s \tag{16}$$

$$v_{c,d,t} = 0 \;\; \forall c \in C, (d,t) \in CA_c \tag{17}$$

$$x, y, v, w \text{ binary} \tag{18}$$

where the objective function in Equation 1 is used to ensure that courses are assigned to good timeslots and can be used to give priority to younger students. The parameter $\alpha_{s,d,t}$ is the cost of assigning student s to a course in timeslot t on day d. Constraints 2, 3, 4, 5, 6 and 7 are used to connect the main variable x to the other variables y, v and w. Constraint 8 ensures that there are no conflicts for the students while Constraint 9 ensures that conflicting courses are not scheduled in the same timeslot, as when a single teacher is teaching multiple courses. Constraint 10 ensures that the number of students assigned to each course is not over capacity. Constraint 11 makes sure that each student is assigned to the correct courses. Constraint 12 handles courses that cover multiple timeslots while Constraint 13 is used to handle courses that must be scheduled more than once during the week. Constraint 14 is used to ensure that the scheduled courses fit into the available classrooms, by limiting the number of courses from each group that can be scheduled simultaneously.

Constraint 15 makes sure that there is at least one day between any two courses that cannot be assigned on consecutive days. Finally, Constraints 16 and 17 handle the availability of students and teachers.

This model does not provide an elegant way of handling isolated sessions, i.e. that the private sessions for a student should be adjacent or close to adjacent to the group courses that the student is assigned to. Once way to handle that problem is to solve the problem of finding a good schedule in two phases, the first phase focuses on group courses while the second phase schedules the private sessions. This two phase method allows us to use the solution from the first phase to update both the availability of students and teachers, as well as adjusting the $\alpha_{s,d,t}$ parameters for the second phase for the timeslots that are close to the group course that the student was assigned to. This approach harmonizes well with the usual manual process where the group courses are scheduled first and then the private sessions, since the schedule for the group courses is considered more important than the private sessions - it is easier to make special arrangements for the private sessions.

4 Case-study - Tónmenntaskóli Reykjavíkur

Our case study comes from Tónmenntaskóli Reykjavíkur [14], a music school in Reykjavík, Iceland. The school focuses on children aged 6 - 16, although children as young as 5 can start in violin preschool. Students are grouped based on age and take group courses in music theory. The youngest children start in flute-preschool, a group course that combines flute and music theory. The students that have finished the flute-preschool can select an instrument and take private lessons on that instrument, as well as attending music theory courses. The school offers the choice of the following instrument: piano, guitar, violin, viola, cello, western concert flute, clarinet, saxophone and bassoon. Students that have reached a certain level of proficiency on their instrument take part in band sessions, such as wind ensemble and string ensemble. Tónmenntaskóli Reykjavíkur uses 30 minutes as the length of each timeslot when scheduling their timetables. Each private session takes one timeslot, group courses such as music theory and preschools take two timeslots, while band sessions can take three to four timeslots.

The case study from Tónmenntaskóli Reykjavíkur is based on the registration in the Fall semester of 2017. The case study contains 124 students registered in a total of 22 group courses and 206 private sessions. These 228 courses and sessions are assigned to 15 teachers, 11 of which focus on private sessions, four teachers handle the group courses and one teacher is teaching both group courses and private sessions.

The school contains 16 classrooms, available at any day and time of the week unless they are being used by the school itself. There are two classrooms that are large enough to be used for group courses and band sessions. There are five classrooms that are preferably used for piano lessons, two for violin lessons while the rest of the classrooms are used for all other private lessons.

The assignement penalties for the student, i.e. the penalty that we incur if we schedule a student to a specific timeslot, was set using the function $pen(t) = \lfloor t/2 \rfloor$ where t is the timeslot. This means that the first two timeslots of each day have no penalties, the next two incur a penalty of one-half, and so on. We would like to be able to set different penalties based on the age of the students, so the younger students get scheduled earlier in the day, but since our case study does not contain any personal information on the students, not even their age, we use the same assingment penalties for all the students.

4.1 Case Study Data Structure

The data for the music school timetabling problem case study is available online [10]. The data does not fit trivially into the XML High School Timetabling format [12] due to differences between the music school and high school timetabling problems, so we store the data using JSON with the following structure:

JSON Structure

- settings
  - D: Number of days
  - T: Number of timeslots
- students (list)
  - id: Integer
  - assignment penalties: Array ($D \times T$) of double numbers
  - unavailable: $[(d_1, t_1), (d_2, t_2), \ldots]$
  - course assignment: $[([c^1_1, c^1_2, \ldots], k_1), ([c^2_1, c^2_2, \ldots], k_2), \ldots]$
- courses (list)
  - id: Integer
  - unavailable: $[(d_1, t_1), (d_2, t_2), \ldots]$
  - max students: Integer
- course info
  - course conflicts: $[(c^1_1, c^1_2), (c^2_1, c^2_2), \ldots]$
  - double sessions: $[(c^1_1, c^1_2), (c^2_1, c^2_2), \ldots]$
  - course pairs: $[(c^1_1, c^1_2), (c^2_1, c^2_2), \ldots]$
  - non consecutive days: $[(c^1_1, c^1_2), (c^2_1, c^2_2), \ldots]$
  - course groups: $[([c^1_1, c^1_2, \ldots], cg_1), ([c^2_1, c^2_2, \ldots], cg_2), \ldots]$

The settings section of the JSON datastructure specifies how many days and timeslots are in our schedule. The students section is a list where, for each student, the JSON structure contains the id of the student, the penalties for each possible timeslot, a list of timeslots where the student is not available, and finally the list of courses that the student should be assigned to. In many cases, such as for the group sections, there can be many courses available and the student should be assigned to a specific number of them. For example, there can be up to four group courses teaching the same material and the student needs to be assigned to one of these courses.

The course section of the JSON datastructure is a list where each course is one element in the list, containing the following information: the id of the course, a list of timeslots where the course cannot be scheduled and the maximum number of students that can be enrolled in the course.

For the course info section, the course conflicts, double sessions, course pairs and non-consecutive days parts contain lists of pairs of courses that fall under each category, while the course groups section contains lists of courses that fall under a specific course groups as well as a counter that specifies how many courses from each course group can be scheduled simultaneously. For our case study, the course groups are group sessions, piano lesssons and violin lessons.

4.2 Case-Study Solution

Every student has been assigned to a teacher beforehand, and most of the band sessions have also already been decided. Since the band sessions are often later in the day or even on Saturdays, the school prefers to schedule them manually. The band sessions are therefore not included in our datastructure, but the availability of the students has been modified to take the pre-scheduled band sessions into account.

We used Gurobi Optimizer version 7.5.2 build v7.5.2rc1 (mac64) [4] as our solver with the model implemented in Python 3.6.2 using the Gurobi Python interface. The computer we used to run the optimization models is a MacBook Pro with a 2.9 GHz Intel Core i5 CPU and 16 GB of memory.

As shown in Figure 1, solving the problem to optimality within the time-limit of 5 hours turned out to be difficult. The full model with both group courses and private lessons had 1,770,568 binary variables, 5,367,717 constraints and 17,606,330 nonzeros. The solver found a feasible solution to the model in 66 seconds, with an objective value of 1246, which was improved to 327 after around 10 minutes of running time. After the time limit of 5 hours, the best solution has an objective value of 310, with an integrality gap of 48%.

The current timetabling process at Tónmenntaskóli Reykjavíkur uses a two-phased approach, where they first create a timetable for the group sessions and then in phase two schedule the private lessons. The group courses are considered more important since it is easier to change the private lessons, and the group courses are also more difficult to schedule. We tried the same approach, using a two-phased approach with only group courses in phase one and private sessions in phase two. Figure 2 shows the value of the incumbent solution and the best lower bound for the first phase of the two-phase solution over the time horizon of 5 hours. The model for the first phase has 286,780 binary variables, 861,547 constraints and 2,794,536 nonzeros. The solver found a feasible solution with objective value of 573 in 26 seconds, and an improved solution with value 174 after around 25 minutes. After 5 hours of running time, the incumbent solution had value of 134 while the best lower bound was 19.1, giving a very high integrality gap of 85.7%.

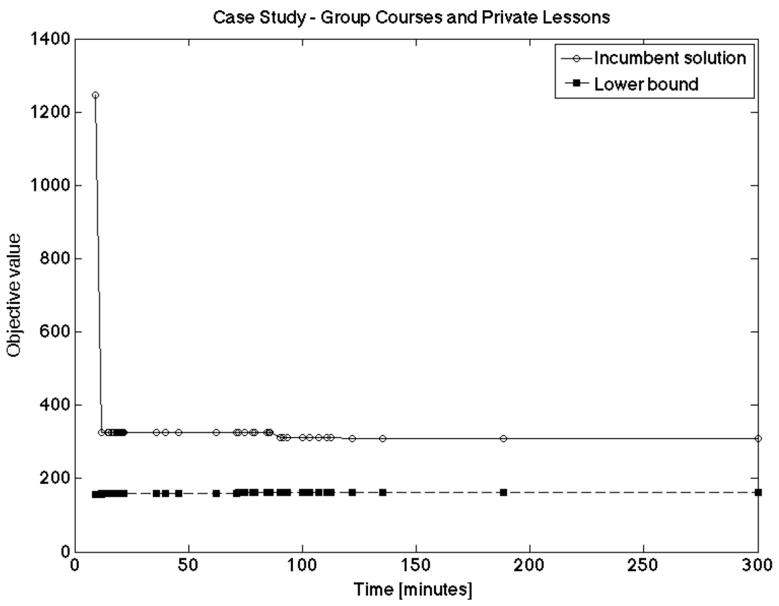

Fig. 1 Case Study - Group Courses and Private Lessons

After the first phase, we updated the availability of the students according to the solution and started on phase two, i.e. scheduling the private lessons. We decided not to change the penalties for the timeslots, for example to make it more likely that private sessions would be close to group courses. By having the same penalties we can compare the solution from the two-phased approach to our inital approach where we attempted to schedule the whole timetable at once. Solving the second phase of our two phased approach was relatively trivial, the solver found the optimal solution of 140 within 15 seconds.

It is very interesting to see that by splitting the problem into two phases, we get a better solution after 5 hours of running time, than if we try to solve the problem as a whole. The best feasible solution we got from attempting to solve the whole problem had an objective value of 310, while using two-phased approach gives a solution with an objective value of $134+140 = 274$. The group courses are the most difficult to schedule, so giving the solver the opportunity of focusing on the group courses without having the private lessons interfering results in a better solution overall. It should be noted that the only objective value we are considering here is a somewhat arbitrary penalty function based on which timeslots are used for each student. Since we are using the same penalty for each student, we have a very high symmetry in the formulation and large sets of solutions with the same objective value, which makes the integer optimization model difficult to solve.

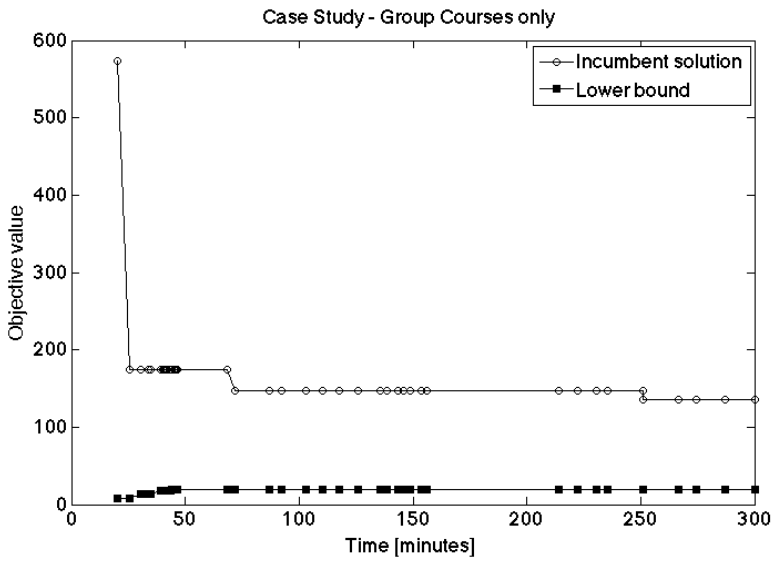

Fig. 2 Case Study - Phase One - Group Courses only

5 Conclusions

The Music School Timetabling Problem is an interesting hybrid between the University Timetabling Problem and the High-School Timetabling Problem. The Music School Timetabling Problem has students in both group courses and private lessons, with restrictions on availability on both teachers and students. We introduced an integer programming formulation for the problem and a case study from Tónmenntaskóli Reykjavíkur. The best approach for solving the case study using integer optimization was to split the problem into two phases, where we start by scheduling all group courses and then the private lessons.

Even though our solver did not manage to find the optimal solution within a time limit of 5 hours of running time, the incumbent solutions were still of very high quality, with feasible timetables where the group courses were scheduled early in the day. The music school timetabling problem has multiple hard constraints, and in our case study the feasible solution did not violate any hard constraints. The initial findings for this problem and our solution methods are promising and the plan is to use our approach to schedule the timetables at Tónmenntaskóli Reykjavíkur in the future.

Acknowledgements The authors would like to thank Rúnar Óskarsson, rector of Tónmenntaskóli Reykjavíkur, for fruitful discussions, data and insights into the problem.

References

1. Babaei, H., Karimpour, J., Hadidi, A.: A survey of approaches for university course timetabling problem. Computers & Industrial Engineering **86**, 43–59 (2015)
2. Bardadym, V.A.: Computer-aided school and university timetabling: The new wave. In: E. Burke, P. Ross (eds.) Practice and Theory of Automated Timetabling, pp. 22–45. Springer Berlin Heidelberg, Berlin, Heidelberg (1996)
3. Burke, E., de Werra, D., Kingston, J.: Applications to timetabling. Handbook of Graph Theory pp. 445–474 (2004)
4. Gurobi optimization. `http://www.gurobi.com/`. Accessed: 2018-02-26
5. K. Burke, E., Jackson, K., H. Kingston, J., F. Weare, R.: Automated university timetabling: The state of the art. The Computer Journal **40**, 565–571 (1997)
6. Kingston, J.H.: A tiling algorithm for high school timetabling. In: E. Burke, M. Trick (eds.) Practice and Theory of Automated Timetabling V, pp. 208–225. Springer Berlin Heidelberg, Berlin, Heidelberg (2005)
7. Kristiansen, S., Stidsen, T.: A Comprehensive Study of Educational Timetabling - a Survey. Department of Management Engineering, Technical University of Denmark (2013)
8. Lewis, R.: A survey of metaheuristic-based techniques for university timetabling problems. OR Spectrum **30**(1), 167–190 (2008)
9. McCollum, B.: University timetabling: Bridging the gap between research and practice. In: Proceedings of the 5th International Conference on the Practice and Theory of Automated Timetabling, pp. 15–35. Springer (2006)
10. Music school timetabling case study: Tónmenntaskóli reykjavíkur. `http://www.ru.is/faculty/eyjo/data/MSTP_data.json.zip`. Accessed: 2018-02-26
11. Pillay, N.: A survey of school timetabling research. Annals of Operations Research **218**(1), 261–293 (2014)
12. Post, G., Ahmadi, S., Daskalaki, S., Kingston, J.H., Kyngas, J., Nurmi, C., Ranson, D.: An xml format for benchmarks in high school timetabling. Annals of Operations Research **194**(1), 385–397 (2012)
13. Schaerf, A.: A survey of automated timetabling. Artificial Intelligence Review **13**(2), 87–127 (1999)
14. Tónmenntaskóli reykjavíkur. `https://www.tonmenntaskoli.is/`. Accessed: 2018-02-26