A mixed-integer linear programming algorithm for final exam scheduling

Szilvia Erdős · **Bence Kővári**

Abstract The automatic generation of schedules has been in the focus of researches for decades. Since small changes in the input have an exponential impact on the tested state space, methods based on heuristics, linear programming, and artificial intelligence are the most successful. Final exam scheduling is a special subtask of the generation of schedules, where special requirements restrict the state space. The problem is examined with an integer linear programming approach. A scoring system is elaborated, wherewith the goodness of the generated schedules is measurable and comparable. The algorithm is tested on an actual test set, which contained the registration of 100 students on the finals of bachelor's degrees. The results show that there is an optimal solution for this complexity. With some improvements on the algorithm, there can be solutions, which are better and fairer than the manually compiled schedules.

Keywords Final exam scheduling · Mixed-integer linear programming · Scheduling algorithm · State examination · Examination timetabling · Operations research · Optimization

1 Introduction

The final examination takes place at the end of the course in most universities. One of the most common forms of this exam is the oral examination, where at one time in one room only one student takes the exam in front of a board of examiners. All these instructors have a special role, and some roles have so high requirements that only a few people can fulfil it. The scheduling of the final exams may be done manually, but the non-automated process can cause human errors, and it is hard to see it all, if all the requirements were fulfilled, and if the scheduling is suitable for everyone. The complexity of this problem is based on two levels.

Szilvia Erdős
Budapest University of Technology and Economics
Department of Automation and Applied Informatics
E-mail: Erdos.Szilvia@aut.bme.hu

Bence Kővári
Budapest University of Technology and Economics
Department of Automation and Applied Informatics
E-mail: Kovari@aut.bme.hu

The scheduling problem is NP-complete [8]. Besides, many final-exam specific conditions must be fulfilled, and some of them contradict each other readily. For example, we want to distribute the workload among the examiners equally, but some instructors may have many more students than others. It is also possible that the instructors (who have to be there on a student's exam) are not available at the same time.

The organization of this extended abstract is as follows. Besides Introduction, the extended abstract features six sections. In Section 2, we overview existing research on the need for oral exams and examination timetabling. Section 3 describes the problem statement and shows the challenges of final exam scheduling. Section 4 presents the structure of the proposed approach and builds up the integer linear programming model for final exam scheduling. In Section 5, the results are evaluated on a real-world data set. Section 6 presents the results and success of the algorithm compared with our previous algorithms and algorithms from the literature made for similar problems. Finally, Section 7 includes concluding remarks.

2 Background

Scheduling is a widely researched topic in literature as several fields in our lives need to be scheduled: preparing a timetable, scheduling our agenda, or scheduling working shifts also belongs to this field.

A large area of research is examination timetabling, which includes the scheduling of exams in universities. This field is the closest to the final exam scheduling problem presented in detail in Section 3.

The goal is to allocate a session and a room to every exam to satisfy a given set of constraints in the general problem. The result is a feasible exam timetable. However, each institution will have some unique combination of constraints, as policies differ from institution to institution. Due to this diversity, the constraints of algorithms in the literature are also various.

For example, in the studies of Wijgers and Hoogeveen (2007) [12], the examination days were fixed, and every student could have only one exam per day. The availabilities of instructors were considered but having an equal workload was not in focus. Al-Yakoob, Sherali, and Al-Jazzaf (2010) [1], in their paper, considered availabilities too. Moreover, they also took into account the distance between buildings. However, they did not handle the exceptional cases like specific instructors could be assigned to given exams and instructors could not have unique roles.

Kochaniková and Rudová, in their article from 2013 [9], gave a solution for oral final exam scheduling, where a student and an instructor are assigned to every exam. They dealt with the availabilities and parallel sessions, but the workloads were not in scope. In addition to the shortcomings of the previously mentioned article, the paper of Ivancevic, Knezevic, and Lukovic (2014) [7] did not take into account the availabilities. However, the instructors were scheduled for whole blocks, and parallel exams were also allowed. Bergmann, Fischer, and Zurheide (2014) [3] also considered the parallelization and workloads too, but instructors could not have particular roles. Aslan, Şîmşek, and Karkacier in 2017 [2] presented an algorithm with equal workloads and collision prevention, but the availabilities of the people were not in scope.

As can be seen from the previous examples, although the same problem is being addressed, the requirements taken into account can vary widely depending on the researchers' priorities.

3 Problem statement

This section shows how the final exam scheduling builds up. A small example is shown in Figure 1, where the schedule of a day is shown.

Day1	Student	Supervisor	Chair	Secretary	Member	Examiner
1	Levente	Hideg Attila	Vajk István	Hideg Attila	Dudás Ákos	Dudás Ákos
2	Tímea	Dudás Ákos	Vajk István	Hideg Attila	Dudás Ákos	Benedek Zoltán
3	Kata	Ekler Péter	Vajk István	Hideg Attila	Ekler Péter	Benedek Zoltán
4	Bence	Kővári Bence András	Vajk István	Hideg Attila	Kővári Bence András	Benedek Zoltán
5	Bence Zsigmond	Ekler Péter	Vajk István	Hideg Attila	Kővári Bence András	Kővári Bence András
6	Péter Szabolcs	Forstner Bertalan	Forstner Bertalan	Pomázi Krisztián	Kővári Bence András	Kővári Bence András
7	Márton	Tóth Tibor	Forstner Bertalan	Pomázi Krisztián	Kővári Bence András	Kővári Bence András
8	Attila	Szabó Gábor	Forstner Bertalan	Pomázi Krisztián	Asztalos Márk	Goldschmidt Balázs
9	Dániel Gábor	Mezei Gergely	Forstner Bertalan	Pomázi Krisztián	Kővári Bence András	Kővári Bence András
10	Gergő	Hamar János Krisztián	Forstner Bertalan	Pomázi Krisztián	Csorba Kristóf	Csorba Kristóf

Fig. 1 Example scheduling for one day

An examination period consists of timeslots (T is the set of all timeslots, $t \in T$ is one of the timeslots). The number of timeslots is equal to the number of students who must take the exam in that semester. The set of students is S. In each timeslot, there is one final exam named e. All exams together compound the whole scheduling named E.

A block is a homogeneous group of students per morning or afternoon, namely half a day, where students are from the same faculty. The set of all blocks is B, and one block is $b \in B$.

The participants and the relationships between them are shown in the Figure 2.

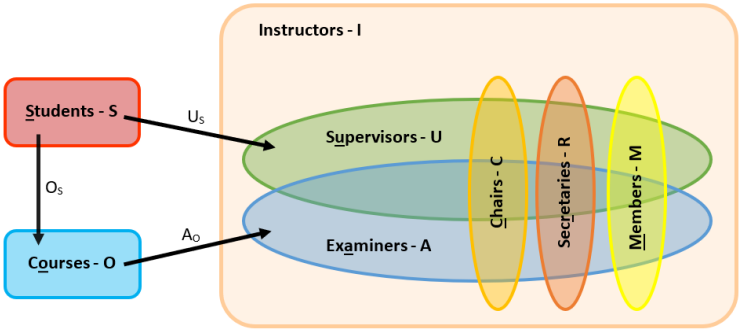

Fig. 2 Participants and their relationships

The instructors are a separate group named I. Several specific subsets can be identified within I:

- The set of supervisors is named $U \subseteq I$. Every $s \in S$ has a specialized $u \in U$, thus $\{s \in S\} \rightarrow \{u \in U\}$. A supervisor may be assigned to several students: $\{u \in U\} \rightarrow \{s_1, s_2, \ldots, s_n \mid \forall\, s_i \in S\}$. Afterwards u_s means the supervisor of student s.
- The group of chairs is named $C \subseteq I$. The regulations of the given final exams specify who can fill this role.
- The group of secretaries is named $R \subseteq I$. They are the rapporteurs of the exams.

– The group of inner members is named $M \subseteq I$. They are the inner instructors of the department where the exam takes place.

There is also a resource group, which consists of courses (noted with O). Every student has to choose a course, thus s student has an exam in course o_s. For each course, it is predefined who can sit the exam for that course. It is defined for every $o \in O$ who can examine from that course. These instructors are examiners named $A_o \subseteq I$, so for every $\{o \in O\} \rightarrow \{a_1, a_2, \ldots, a_n \mid \forall\, a_i \in A_o\}$. Afterwards, $\bigcup_{o \in O} A_o = A$

An exam stands up, as shown in equation 1.

$$e \in \{t, s, u, c, r, m, a \mid t \in T,\ s \in S, u \in U, c \in C, r \in R, m \in M, a \in A_o, o \in O_s\} \tag{1}$$

4 The proposed mixed-integer linear programming model

This section discusses how the mixed-integer linear programming model builds up for this unique timetabling problem – for the final exam scheduling.

Integer linear programming is a well-known method for optimization problems[4]. The standard form of it can be seen henceforth: there are some integer decision variables, which values are searched, while the linear objective function is optimized, and linear equality and inequality constraints are subjected.

4.1 Decision variables

By choosing the variables, they must be as suitable as they can to the actual problem. Besides, there should be no unnecessary variables to avoid redundant calculations.

There are two primary variables, and both are binary. One is for instructors, and the other represents the students. Both variables have two dimensions: people and timeslots. If the value of $i \in I$ instructor in timeslot $t \in T$ is 0, then instructor i is not scheduled in t timeslot, and if this value is 1, then i is scheduled in t. The same holds for students. The decision variables of instructors are $x_{i,\,t} \in \{0,1\} \mid \forall i \in I,\ \forall t \in T$, and the variables of students are $x_{s,\,t} \in \{0,1\} \mid \forall s \in S,\ \forall t \in T$.

The instructors, who have special roles, are noted as follows. The instructor who can be a chair on an exam is shown as $x_{c,\,t}$, which is identical to the variable $x_{i,\,t}$, where $i \in C$ holds.

Some additional decision variables are necessary for the fulfilment of certain constraints. These were calculated based on the primary decision variables.

A hard requirement of the final exam scheduling is that the chairs and the secretaries must be scheduled in whole blocks. Additional binary variables were introduced that present the scheduling blocks of chairs and secretaries. These are shown in equation 2.

$$\begin{aligned} x_{c,\,b} &\in \{0,1\} \ \ \forall c \in C,\ \forall b \in B \\ x_{r,\,b} &\in \{0,1\} \ \ \forall r \in R,\ \forall b \in B \end{aligned} \tag{2}$$

Furthermore, some variables were constructed for optimizing the workload of instructors. The main point of this is to determine the difference between the optimal and the actual workloads. However, in linear programming, there is no simple way to express the absolute value of two decision variables. A method is to introduce two additional variables for one person. The workload of chairs, secretaries, and members is optimized by applying the variables like in formula set 3.

$$\begin{array}{llll} x_c^\alpha \in \mathbb{Z}^+ & \forall c \in C & x_r^\alpha \in \mathbb{Z}^+ & \forall r \in R \quad x_m^\alpha \in \mathbb{Z}^+ \quad \forall m \in M \\ x_c^\beta \in \mathbb{Z}^+ & \forall c \in C & x_r^\beta \in \mathbb{Z}^+ & \forall r \in R \quad x_m^\beta \in \mathbb{Z}^+ \quad \forall m \in M \end{array} \tag{3}$$

4.2 Objective function

Final exam scheduling has to fulfil some soft requirements (like instructors' optimal workload or availabilities). These requirements are primarily defined in the objective function, which looks like the expression 4.

$$min \sum_{i}^{i \in I} \sum_{t}^{t \in T} (x_{i,t} * Cost_{i,t}) + \sum_{c}^{c \in C} \left(x_c^\alpha + x_c^\beta \right) + \sum_{r}^{r \in R} \left(x_r^\alpha + x_r^\beta \right) + \sum_{m}^{m \in M} \left(x_m^\alpha + x_m^\beta \right) \tag{4}$$

The first part is for the availabilities of instructors where $Cost_{i,t}$ is a positive integer constant, which belongs to the penalty point of instructor $i \in I$ if he or she is not available in timeslot $t \in T$.

The rest of the objective function belongs to the requirements of having equal workloads for chairs, secretaries, and members. Here the derived decision variables of each instructor in these roles are summarized, where for example, by the chairs x_c^α means the difference from the optimal workload in the positive direction (how much more than optional scheduled exams the chair c has), and x_c^β means the difference in the negative direction. Of course, at least one of these two variables have to be null.

4.3 Problem constraints

The scheduling has to fulfil many different requirements, and most of them are defined in various ways as linear equality and inequality constraints. There are three types of problem constraints which are defined below.

4.3.1 Constraints for derived decision variables

The first type of constraint refers to the derived decision variables. This formula set contains the binary variables that present the scheduling blocks of chairs (shown in equation 5) and secretaries (shown in equation 6). These variables have the value of 1 only if the decision variables in every timeslot in that given block are 1 – this means that the instructor is scheduled in the whole block. The condition was formulated using "logical and" operations, taking advantage of the solver's capabilities. (Gurobi was used in the modelling, but many other solvers can map these operations to MILP conditions.)

$$x_{c,b} = \bigwedge_{t \in b} x_{c,t} \qquad \forall c \in C \, , \, \forall b \in B \tag{5}$$

$$x_{r,b} = \bigwedge_{t \in b} x_{r,t} \qquad \forall r \in R \, , \, \forall b \in B \tag{6}$$

The other derived variables are for optimizing the workload of instructors. In the case of chairs, formula 7 should be minimized , where D_c is the optimal value of the workload of a chair.

$$\left| \sum_{b}^{b \in B} x_{c,b} - D_c \right| \tag{7}$$

However, as stated before, the absolute value could not be placed in the objective function, so instead of it, there should be two derived variables x_c^{α} and x_c^{β}, subject to equation 8.

$$x_c^{\alpha} - x_c^{\beta} = \sum_{b}^{b \in B} x_{c,b} - D_c \qquad \forall c \in C \tag{8}$$

Furthermore, in the objective function there is $x_p^{\alpha} + x_p^{\beta}$ instead of the absolute value 7.

The constrains for secretaries (9) and members (10) are similar. D_r is the optimal workload of secretaries and D_m refers to the optimal workload of members.

$$x_r^{\alpha} - x_r^{\beta} = \sum_{b}^{b \in B} x_{r,b} - D_r \qquad \forall r \in R \tag{9}$$

$$x_m^{\alpha} - x_m^{\beta} = \sum_{t}^{t \in T} x_{m,t} - D_m \qquad \forall m \in M \tag{10}$$

4.3.2 Constraints refer to the basis of scheduling

Many constraints are necessary for getting an adequate final exam scheduling. The list of them can be found in Table 1.

Table 1: Constraints for the fundamental contribution of scheduling

$\sum_i x_{i,t} \leq 5 \quad i \in I, \forall t \in T \qquad (11)$	There should be a maximum of 5 instructors in a timeslot.
$\sum_c x_{c,t} = 1 \quad c \in C, \forall t \in T \qquad (12)$	There should be one instructor at each timeslot who can be the chair.
$\sum_r x_{r,t} = 1 \quad r \in R, \forall t \in T \qquad (13)$	There should be one instructor at each timeslot who can be the secretary.
$\sum_m x_{m,t} \geq 1 \quad m \in M, \forall t \in T \qquad (14)$	There should be at least one instructor at each timeslot who can be the member. There are some situations when there must be more instructors in an exam who could be the member (e.g. the supervisor and the examiner could also be members and different people). That is the reason for the minimum criteria.
$\sum_m x_{m,t} \leq 2 \quad m \in M, \forall t \in T \qquad (15)$	There should be a maximum of two instructors in a timeslot who could be in the role of member.

$\sum_s x_{s,t} = 1 \quad s \in S,\ \forall t \in T$	(16)	There should be precisely one student in every timeslot.
$\sum_t x_{s,t} = 1 \quad t \in T,\ \forall s \in S$	(17)	Every student should be in exactly one timeslot.
$x_{s,t} - x_{u_s,t} \leq 0 \quad \forall t \in T\ ,\ \forall s \in S$	(18)	The supervisor of the student should be there on the exam when the student has the examination.
$x_{s,t} - \sum x_{a_{c_s},t} \leq 0 \quad \forall t \in T\ ,\ \forall s \in S$	(19)	An examiner from the student's course should be there on the exam when the student has the examination.

4.3.3 Constraints for further requirements

The model has to fulfil all the hard requirements which were defined before. These are listed in Table 2.

Table 2: Constraints for further hard requirements

$\sum_c x_{c,b} = 1 \quad c \in C,\ \forall b \in B$	(20)	There should be precisely one chair in every block.
$\sum_r x_{r,b} = 1 \quad r \in R,\ \forall b \in B$	(21)	There should be precisely one secretary in every block.
$\sum_c^{c \in C} \sum_t^{t \in T} (x_{c,t} * Cost_{c,t}) = 0$	(22)	Chairs should be scheduled if only if they are available. ($Cost_{c,t}$ is the penalty score for non-availability.)
$\sum_r^{r \in R} \sum_t^{t \in T} (x_{r,t} * Cost_{r,t}) = 0$	(23)	Secretaries should be scheduled if only if they are available. ($Cost_{r,t}$ is the penalty score for non-availability.)
$\sum_b^{b \in B} x_{c,b} \geq D_c^- \quad \forall c \in C$ $\sum_b^{b \in B} x_{c,b} \leq D_c^+ \quad \forall c \in C$	(24)	The workload of chairs should be held between limits. (D_c^- is the absolute minimum value of the workload of chairs, D_c^+ is the maximum.)
$\sum_b^{b \in B} x_{r,b} \geq D_r^- \quad \forall r \in R$ $\sum_b^{b \in B} x_{r,b} \leq D_r^+ \quad \forall r \in R$	(25)	The workload of secretaries should be held between limits.
$\sum_t^{t \in T} x_{m,t} \geq D_m^- \quad \forall m \in M$ $\sum_t^{t \in T} x_{m,t} \leq D_m^+ \quad \forall m \in M$	(26)	The workload of members should be held between limits.

5 Case study

The algorithm was tested on an actual test set, acquired from the Budapest University of Technology and Economics (BME), Department of Automation and Applied Informatics. The official regulation of BME is available online at [10] and [11]. The test set contains 100 students, 49 instructors and 12 courses. Just for these 100 students, the number of possible schedules is about 10462, according to equation 27.

$$100! \times 4^{100} \times 9^{100} \times 10^{100} \times 3^{100} \approx 10^{462} \tag{27}$$

where 100 is the number of timeslots, 4 is the number of chairs, 9 is the number of secretaries, 10 is the number of members, and 3 is the average number of instructors for a course.

As noted before, the Gurobi solver was used to solve the model.

There were 15106 decision variables, 260 constraints, and it runs for $1,3$ seconds on 8 threads. The minimized objective value was 40.

We analysed the solution by hand, and we found out that members and secretaries did not fulfil only requirements for equal workloads. The only reason for this was that these instructors have many more students than the others, and they should be there on more exams as supervisors.

6 Achieved results

We have made a scoring system in advance for requirements, which is practical for measuring the schedule's goodness. Furthermore, the schedules could be compared to each other based on the penalty points. We have made two different algorithms before for the final exam scheduling, a genetic algorithm-based [6] and a heuristic approach based on pair graphs and the Hungarian method [5].

First, we compared the results of MILP with our own earlier algorithms. The outcomes are illustrated in Figure 3. As it can be seen, the MILP-based algorithm was the best in every sight. MILP scores more than ten times better than the previous best-performing heuristic algorithm and performs better orders in runtime under similar conditions.

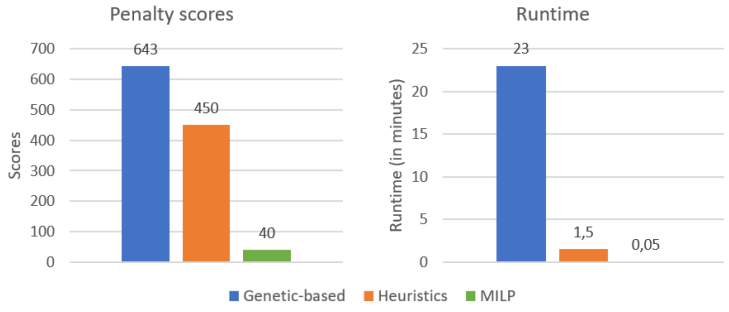

Fig. 3 The comparison of own scheduling algorithms

MILP models are often worth examining from a scalability point of view. In final exam scheduling, 100 students are the average number of students per student group. There may be some increase for popular courses (such as computer science, for which the test case was made). However, given the university's capacity, the maximum number of students per teaching base is 150. Taking this case into account, the performance of the algorithm was tested for both 125 and 150 students, and a fictitious set of 175 and 200 students were generated, which far exceeds the real numbers. The results of this are shown in Figure 4, which shows that for 125 students, there is little difference, while above 150 students, the runtime starts to increase. However, it can be seen that for a set of 200 students, the running time is significantly better than for any other algorithm on a set of 100 students.

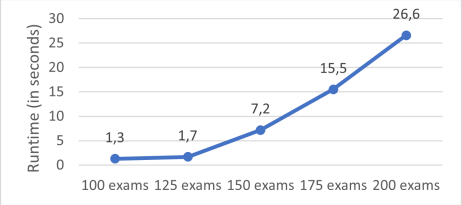

Fig. 4 Testing the scalability of the algorithm

Furthermore, the results are compared to some algorithms discussed in the literature (close to our problem's statement) based on comprehensive properties. The results are depicted in the Table 3, where ✓ means that this requirement is considered and fulfilled, and ✗ otherwise. ✓/✗ means that it is invented but not implemented yet.

Table 3 Algorithms compared along with their abilities

	requirements not equally important	equal workload of instructors	availabilities of people	block-based structure	parallel timeslots and collision handling	specific roles of instructors
own MILP-based algorithm	✓	✓	✓	✓	✓/✗	✓
Wijgers, Hoogeveen [12] (2007)	✗	✗	✓	✓	✗	✗
Al-Yakoob, Sherali, Al-Jazzaf [1] (2010)	✓	✓	✓	✗	✓	✗
Kochaniková, Rudová [9] (2013)	✓	✗	✓	✗	✓	✓
Bergmann, Fischer, Zurheide [3] (2014)	✓	✓	✓	✗	✓	✗
Ivancevic, Knezevic, Lukovic [7] (2014)	✗	✗	✗	✓	✓	✓
Aslan, Şîmşek, Karkacier [2] (2017)	✗	✓	✗	✗	✓	✗

According to these, it can be said that our algorithm gives a more comprehensive solution for more questions. It covers more abilities of final exam scheduling than the algorithms discussed before in the literature for similar problems.

7 Conclusion

Despite the popularity of scheduling, the topic of the final exam scheduling provides another exciting problem because it cannot be created in a ”normal” scheduling design methodology. It needs comprehensive solutions.

A model based on linear programming has been developed, which considers more aspects at the same time than similar solutions discussed in the literature. It also far outperforms algorithms previously developed for this problem.

With some improvements to our algorithm (like introducing parallel exams), there can be solutions, which are better and fairer than the manually compiled schedules.

Acknowledgements Project no. FIEK_16 – 1 – 2016 – 0007 has been implemented with the support provided from the National Research, Development and Innovation Fund of Hungary, financed under the Centre for Higher Education and Industrial Cooperation - Research infrastructure development (FIEK_16) funding scheme.

References

1. Al-Yakoob, S., Sherali, H., Al-Jazzaf, M.: A mixed-integer mathematical modeling approach to exam timetabling. Computational Management Science **7**, 19–46 (2010). DOI 10.1007/s10287-007-0066-8
2. ASLAN, E., ŞİMŞEK, T., KARKACIER, A.: A binary integer programming model for exam scheduling problem with several departments. Bilgi Ekonomisi ve Yönetimi Dergisi **12**(2), 169–175 (2017)
3. Bergmann, L.K., Fischer, K., Zurheide, S.: A linear mixed-integer model for realistic examination timetabling problems. In: Proceedings of the 10th International Conference on the Practice and Theory of Automated Timetabling, pp. 82–101 (2014)
4. Dantzig, G.: Linear programming and extensions. Princeton university press (2016)
5. Erdos, S.: Algorithm based on Hungarian Method for Final Exam Scheduling. In: Automation and Applied Computer Science Workshop (2019)
6. Erdos, S., Kovari, B.: Genetic algorithm based solution for final exam scheduling. In: MultiScience - microCAD International Multidisciplinary Scientific Conference (2019)
7. Ivančević, V., Knežević, M., Luković, I.: A course exam scheduling approach based on data mining. Smart Digital Futures 2014 **262**, 132 (2014)
8. Kirkpatrick, D.G., Hell, P.: On the completeness of a generalized matching problem. In: Proceedings of the Tenth Annual ACM Symposium on Theory of Computing, STOC ’78, p. 240–245. Association for Computing Machinery, New York, NY, USA (1978). DOI 10.1145/800133.804353. URL https://doi.org/10.1145/800133.804353
9. Kochaniková, B., Rudová, H., et al.: Student scheduling for bachelor state examinations. In: Proceedings of the 6th Multidisciplinary International Scheduling Conference–MISTA, pp. 762–766. Citeseer (2013)
10. Rektori Kabinet Oktatási Igazgatóság: A Szenátus X./10./2015-2016. (2016. VII. 11.) számú határozata A BME TANULMÁNYI ÉS VIZSGASZABÁLYZATÁRÓL. http://www.kth.bme.hu/document/2061/original/BME (2016)
11. Sujbert, L.: BME VIK BSc szakdolgozat, záróvizsga, oklevél szabályzat. https://www.vik.bme.hu/document/1343 (2017)
12. Wijgers, R., Hoogeveen, J.: Solving the examination timetabling problem (2007)