Examination timetable modelling at the University of Iceland

Ásgeir Örn Sigurpálsson

Faculty of Industrial Engineering,
Mechanical Engineering and Computer Science
University of Iceland
2017

EXAMINATION TIMETABLE MODELLING AT THE UNIVERSITY OF ICELAND

Ásgeir Örn Sigurpálsson

30 ECTS thesis submitted in partial fulfillment of a
Magister Scientiarum degree in Industrial Engineering

Advisors
Tómas Philip Rúnarsson
Hreinn Pálsson

Examiner
Eyjólfur Ingi Ásgeirsson

Faculty of Industrial Engineering,
Mechanical Engineering and Computer Science
School of Engineering and Natural Sciences
University of Iceland
Reykjavik, April 2017

Examination timetable modelling at the University of Iceland
30 ECTS thesis submitted in partial fulfillment of a M.Sc. degree in Industrial Engineering

Faculty of Industrial Engineering,
Mechanical Engineering and Computer Science
School of Engineering and Natural Sciences
University of Iceland
Dunhagi 5
107, Reykjavik, Reykjavik
Iceland

Telephone: 525 4000

Bibliographic information:
Ásgeir Örn Sigurpálsson, 2017, Examination timetable modelling at the University of Iceland, M.Sc. thesis, Faculty of Industrial Engineering,
Mechanical Engineering and Computer Science, University of Iceland.

Printing: Háskólaprent, Fálkagata 2, 107 Reykjavík
Reykjavik, Iceland, April 2017

Abstract

In this thesis, the examination timetabling will be modelled for the University of Iceland (U.I). As the situation has been, the timetabling problem has been solved manually taking $5-7$ days each semester. By making models that solve the problem could not only lead to less time spent in timetabling but could potentially give a better solution for the students. A two-phase approach was chosen for the problem where Phase I schedules the examinations into slots while Phase II schedules the examinations to suitable rooms. The key objective of Phase I was to make a conflict free timetable but at the same time minimize same day examinations for students, students facing two examinations within 24 hours and students not receiving one day off before an exam. Phase II had multiple objectives but the overall goal was to minimize the rooms and buildings being used for each exam. The results achieved by using the models gave promising results for both phases. The main drawback was the computational time for Phase I. Therefore, a low-level heuristic was proposed. The solution achieved by the heuristic approach gave slightly worse results but required less computational time. By using the heuristic approach, Phase I was split into two sub-phases but by splitting a problem into two sub-phases can lead to an infeasibility. Therefore, a new way must be proposed in the future using a curriculum based scheduling.

Útdráttur

Í þessari ritgerð verða tvö líkön gerð fyrir Háskóla Íslands (HÍ) sem leysa próftöflugerð háskólans. Á undanförnum árum hefur vandamálið verið leyst handvirkt og tekur alla jafna $5 - 7$ vinnudaga að leysa hvert próftímabil. Með innleiðingu slíkra líkana gæti hlotist mikill ávinningur í mögulegum vinnusparnaði og ekki síst fyrir nemendur sem gætu fengið betri próftöflur. Tveggja fasa lausnaraðferð var beitt þar sem Fasi I raðaði prófum í lotur meðan Fasi II notaði lausn frá Fasa I og raðaði prófum í viðeigandi stofur. Við röðun prófa í lotur voru grunngildi röðunarinnar að hafa sem fæsta nema í prófi samdægurs, í röð (*þeas innan* 24 *klukkustunda*) og að nemendur fengu að lágmarki einn dag í hvíld fyrir hvert próf. Á sama tíma voru engir beinir árekstrar leyfðir. Fasi II hafði aftur á móti nokkur markmið en grunngildin voru að nota sem fæstar stofur og byggingar fyrir hvert próf. Að nota líkönin til úrlausnar á próftöflugerðinni gaf góðar niðurstöður fyrir báða fasa. Helsti ókosturinn við notkun líkanana var helst bundinn við Fasa I vegna mikils reiknitíma. Þar af leiðandi var brjóstvitsaðferð notuð þannig að Fasa I var skipt í tvo undirfasa og leyst. Niðurstaða þess gaf örlítið verri niðurstöðu röðunnar en krafðist minni reiknitíma. Notkun brjóstvitsaðferða þar sem einum fasa er skipt upp í tvo undirfasa getur leitt til ófýsileika. Þar af leiðandi væri áhugavert að kanna að nota námsbrautir í framtíðinni við gerð próftöflunnar frekar en að raða hverju og einu námskeiði.

Contents

List of Figures

List of Tables

Notations

$\mathcal{C}$: The set of all courses that should be examined.

$\mathcal{C}_s$: The set of all courses that include special students that should be examined.

$\mathcal{C}_c$: The set of courses that requires being examined in computer rooms.

$\mathcal{C}_m$: The set of courses that do not require any room for their examination.

$\mathcal{E}$: The set of available time slots in the examination period.

$\mathcal{E}_s$: Used to split the room scheduling into different sub-problems rather than solving all slots at once.

F_c^f: The set of time slot/s that course c requires.

$\mathcal{E}_d$: The set of all available days within time slots $\mathcal{E}$.

$\mathcal{H}_e$: A binary indicator if time slots e have a free day before.

d_c: The set of duration of each examination c.

A_{c_1,c_2}: A symmetric incidence matrix of courses c_1 and c_2.

M_{c_1,c_2}: A set of courses that have questions in common in the same time slot.

L_{c_1,c_2}: A set of courses that should be examined in the same time slot.

S_c: The total number of students enrolled to course c.

S_c^s: The total number of special students enrolled to course c.

Capacity

M_e^t : The total number of seats for general students available in each time slot e.

M_e^s : The total number of special seats available in each time slot.

M_c : The total number of computer seats available in each time slot.

M_c^s : The total number of special computer seats available in each time slot.

M_h : The total number of seats available at Stakkahlid (home of School of Education).

Rooms and Buildings

$\mathcal{R}$: The set of all available rooms.

$\mathcal{R}_g$: The set of all available rooms for general students.

$\mathcal{R}_s$: The set of all available special rooms for courses that include special students.

$\mathcal{R}_c^g$: The set of all available computer rooms for general students.

$\mathcal{R}_c^s$: The set of all available computer rooms for special students.

$\mathcal{R}_r$: The capacity of each room r.

$\mathcal{B}$: The set of all available buildings.

$\mathcal{V}_b^r$: The set of all rooms r in buildings b.

$\mathcal{P}_b^c$: A set of the preferred building b for each course c.

$\mathcal{R}_b^c$: A set of required buildings b required by course c.

$\mathcal{O}$: The set of all available clusters.

$\mathcal{Q}_o$: The set of all buildings b in each cluster o.

Variables

$x_{c,e}$: Binary decision variable equal to 1 if course c occupies time slot e.

P_{c_1,c_2}: Variable that indicates if course c_1 will not have a day off before course c_2.

I_{c_1,c_2}: Variable that indicates if courses c_1 and c_2 have exams the same day.

U_{c_1,c_2}: Variable that indicates if courses c_1 and c_2, have examinations in the after-

noon and the morning after.

J: Soft objective variable that indicates the maximum number of students having computer exams for any time slots.

Z: Soft objective variable that indicates the smallest number of students assigned to every time slot.

$h_{c,r}$: Variable that indicates how many students in course c are assigned to room r.

$h^f_{c,r}$: Constant that indicates how many students in course c are fixed to room r.

N_c: Soft objective variable that indicates the number of buildings assigned to course c.

$w_{c,r}$: Binary decision variable equal to 1 if course c occupies room r.

$wb_{c,b}$: Binary decision variable equal to 1 if course c occupies building b.

$wbb_{c,b}$: Binary decision variable equal to 1 if course c occupies building b. The difference between this binary decision variable and $wb_{c,b}$ is that the general examination is not included in this variable.

wr_r: Binary decision variable equal to 1 if a room is occupied else it will tend to 0.

Constants and tolerances

θ: The number of common students of courses c_1 and c_2 allowed to violate hard constraints in the model.

ϕ: The number of common students in courses c_1 and c_2 that are allowed to not have a day off before an exam.

α: An indicator of what a large course is.

μ_r: A set of priority coefficient for each room r.

Acknowledgments

I want to thank my advisor Tómas Philip Rúnarsson for excellent guidance throughout this process. All of the brainstorming sessions were really helpful and memorable and I will be forever thankful for his support and help. Secondly, I would like to thank my co-advisor, Hreinn Pálsson for excellent guidance, countless meetings and all of his faith in the project which was really important since he is the director of examinations for the University of Iceland. It was an honour that the work proposed in this thesis was used to make the official examination scheduling for the fall of 2016 and was again used for the spring of 2017.

I want to thank Hörður Guðmundsson and Ragnar Stefán Ragnarsson, employees of Reiknistofnun Háskóla Íslands for all their work and meetings regarding Ugla. When this was written, the examination timetabling models were being more and more implemented to Ugla in order to make the usage of the models easier in the future. Similarly, I want to thank my fellow student, Jóhann Haraldsson for his help and pre-work on the database.

I want to thank my friends and colleagues for their support and good discussions on my work. I want to thank the University of Iceland for funding this project in the summer of 2016 and making this adventure possible.

Finally, I want to thank my family for endless support over the years.

1. Introduction

In general, our life is full of schedules. We must attend school, to gym, to work etc. and make it all fit in our daily schedule. Scheduling is in fact, a term over multiple things. It can span all from timetable scheduling to workforce scheduling among other different types of schedules. The main focus of this thesis will be on examination timetabling.

Examination timetabling belongs to a class of university timetabling problems but these topics are well researched (Qu et al., 2009). The main task of the examination timetabling is to assign courses to prefixed time slots of equal or varying lengths (Müller, 2016). Each course is assigned to only one time slot that fits its duration. At the same time, each course must fit into a room or rooms such that all of the students assigned to the course have a seat (Kahar and Kendall, 2010; Qu et al., 2009; Leong and Yeong, 1990). Such requirements are classified as hard constraints. Typically, each model consists of hard and soft constraints. Hard constraints cannot be violated, while soft constraints can. Hence, in order to make a good quality solution, the soft constraints must be satisfied as often as possible. (Qu et al., 2009) Other requirements might be room requirements, teachers availability etc. Usually, each university differs from another regarding the requirements (Burke et al., 1996). Although this task may seem effortless to solve, it rapidly gets complicated as the requirements grow (Laporte and Desroches, 1984). The main purpose of making examination timetabling models is first and foremost to make a good examination timetable for students by having as much time between exams as possible for each student, to use as few rooms as possible for the room scheduling and to decrease the manual work. The examination timetabling at the University of Iceland has been solved manually, taking 5 to 7 work days on average each semester.

The course timetabling at the University of Iceland is still done manually. Recently, developments on a Mixed Integer Programming (MIP) model solving such problem was designed for the School of Engineering and Natural Sciences. That work has brought interest in solving the examination timetable problem by a similar model.

In this thesis, two MIP models will be formulated for the examination timetabling for the University of Iceland. The problem will be split into two sub phases. Phase I will be the examination scheduling, where the main goal is to minimize same day examinations, examinations within 24 hours and the number of students that do

not receive a day off before an exam. At the same time, the computer examinations will be spread as evenly throughout the examination period and the assignments of students to each time slot be as equal as possible. In Phase II, the assignments of exams into rooms take place, where the main objective will be to minimize the total number of rooms and buildings used and to assign as many exams into their preferred or required buildings. Similarly, if an exam must be split between rooms it will be assigned to as few buildings as possible and preferably to buildings close by. Since Phase I is computationally expensive, a low level heuristics will be proposed in order to make the computing less expensive.

The program has been given the name Bessý. Bessý is an Icelandic female nickname but the main idea behind that name is it's similarity to the Icelandic word "Bestun" which can be translated to the word optimization.

1.1. Motivation

The main motivation for the work in this thesis is that at the University of Iceland the examination timetabling is still solved manually by the director of examinations. Making models that would solve the timetabling would not only save a lot of time each year spent in timetabling but could potentially improve the quality of the solutions but researchers have revealed great improvements by using algorithms and/or models solving the examination timetabling. Solving this manually is not only time consuming but also a complicated task to perform and typically requires years of training.

One might think that implementation of pre-existing software that exist on the market would solve the situation at University of Iceland. In fact, existing solutions could help to solve the problem but at the same time, there are limitations to such software. Each university differs from another regarding the requirements and constraints that must be used (Arbaoui et al., 2015; Burke et al., 1996; Kahar and Kendall, 2010). Therefore, a special formulation for each university is necessary.

In the spring semester of 2016, I had the opportunity of being a teaching assistant on the course Operation Research. The final project of the course was a more simplified version of the examination scheduling for the School of Engineering and Natural Sciences. What grabbed my interest most, was how different opinions were amongst students in the course on what a good examination scheduling was. Consequently, the models for the examination scheduling was designed with commonly agreed ideas from the students on what a good examination timetable was.

1.2. Objective and contributions

The objective of this study is to formulate models that can be used for the examination timetabling at the University of Iceland. Two MIP models will be proposed where Phase I of the model will assign exams to slots while Phase II will assign the exams to rooms. For Phase I, same day examinations, examinations within 24 hours, students not receiving one day off before an exam will be minimized. Similarly, the total number of students assigned to each time slot will be as equal as possible and the computer based examination will be spread as equally. Phase II will conversely, assign the exams into suitable rooms by minimizing the overall room usage while assigning as many exams to their preferred or required buildings. Another objectives are corresponding if an exam needs to be split into multiple rooms, that it will be split to as few buildings as possible and preferably close by. Lastly, a low level heuristic will be proposed for Phase I in order to make the problem less computationally expensive. Solving the examination timetabling problem is in general computationally expensive since the problem is NP-complete and is therefore rather complicated to solve. Consequently, in this study the main research questions will be the following:

1. What has the most impact on the MIP solver of solving the examination scheduling?

2. What has the most impact on the difficulty of the room scheduling?

3. Is the solution achieved by the low level heuristic a suitable solution?

1.3. Overview

The thesis will be structured in the following way. Chapter 2 provides a general description of the examination timetabling and a literature review. In Chapter 3, a problem description will be given for the University of Iceland and the model design takes place. Similarly, the data will be collected for the problem, transformed to the right format and a brief description will be given on the functionality of the systems which will be proposed. Chapter 4 will present all of the data analysis of the data collected. In Chapter 5 the formulation of the MIP models takes place for both Phase I and Phase II. In Chapter 6, computational experiments will be performed on the models and the results will be analyzed. Chapter 7, will propose a low level heuristic method for Phase I where the problem will be split into two subphases, Phase I-a and Phase I-b due to a high computational time for a one phase approach. The heuristic approach will be analyzed and compared to the original

method. Chapter 8, will present discussions of the work performed. Lastly, Chapter 9, will present the conclusions of the thesis and give suggestions on future work.

2. Background

In this section, a brief overview of the literature previously published on the examination timetabling will be presented. Since 1960, there have been many articles published about educational timetabling by researchers in the field of Operation Research and Artificial Intelligence. Although this topic is well researched, increased research level in this field has been shown (Qu et al., 2009). This chapter will be divided into few sections. In Section 2.1, a general description will be given on what educational timetabling is and how it can be classified. Following that, the examination timetabling problem will be introduced in Section 2.1.1. In Sections 2.2 and 2.3 a literature review of the articles published about the examination scheduling and room scheduling respectively will be presented. Lastly, Section 2.4 presents the most commonly used constraints for the examination timetabling.

2.1. Educational timetabling

One may ask what timetabling is in general. A formal definition of what a timetable is can be found in Burke et al. (2004) paper:

Definition 2.1.1 *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; 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 a possible.*

Even though the definition is a pretty formal statement of the problem, it works for all kind of timetabling including the educational timetabling which is in general to assign time (*slots*) to courses (*resources*) and to satisfy constraints. Constraints that might be put used the timetabling can be either soft or hard. When a constraint is classified as hard, it cannot be violated. An easy example would be when one student is facing two exams at the same time. Obviously, it is impossible for the student to do so and therefore that kind of constraints are classified as hard. Soft constraints differ from the hard constraints. Even though they are violated, they can still generate a feasible solution. A typical example of a soft constraint is the

case when large courses are put in the beginning of the examination period to allow the maximum time for marking (Qu et al., 2009). That occasion is not a must for the problem but good to have.

Normally, the educational timetabling is classified into two main categories: *Examination timetabling* and *Course timetabling*. The course timetabling is the general assignment of time slots and rooms to courses. The course timetabling is not covered in this thesis and therefore T. Birbas (1997) paper is recommended for further knowledge. The examination timetabling is the second group of the educational timetabling. It consists of three sub-categories: *Invigilation staff scheduling*, *Examination scheduling* and *Room scheduling*. In Figure 2.1, the classification of the educational timetabling can be seen.

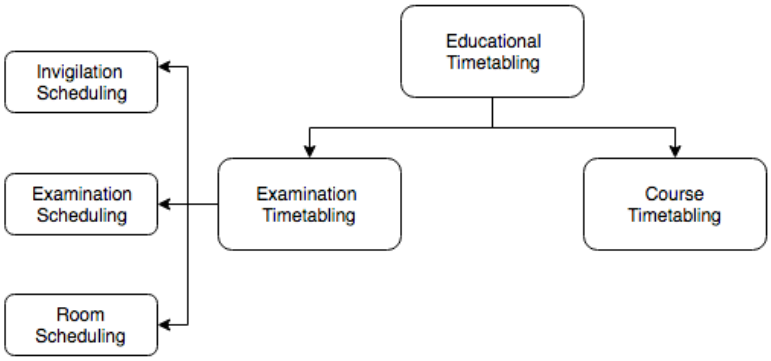

Figure 2.1: Educational Timetabling

2.1.1. Examination timetabling

As mentioned in Section 2.1, the examination timetabling consist of three sub-categories or *Invigilation staff scheduling*, *Examination scheduling* and *Room scheduling*. In this section, a short description of each group will be presented. All of those sub-groups have impacts on another although they can be solved independently.

- **Invigilation staff scheduling**
  Often, staff is needed to proctor the examinations to service students and to prevent cheating. Therefore, invigilation staff must be added to shifts when the examinations take place. As more rooms are added, more staff is needed and therefore assigning the minimum number of rooms is essential (Al-Yakoob et al., 2007)

- **Examination scheduling**
  Examination scheduling is when exams are assigned into time slots so no students are assigned to two exams in the same time slot. Usually, more restrictions are required such that a student should not be facing consecutive examinations i.e. afternoon and morning after or the same day examinations and others. Similarly, soft conditions might be put into use such as large exams in the beginning of the examination period (Qu et al., 2009; Broder, 1964; Cole, 1964).

- **Room scheduling**
  Room scheduling is when exams are assigned to rooms. Often there are guidelines on how the exams should be assigned to rooms. Often, this process is solved by a separated model from the examination scheduling but sometimes they are solved together in a one phase approach (Dammak et al., 2006; Kahar and Kendall, 2010).

A more detailed overview on the examination scheduling and the room scheduling can be found in Sections 2.2 and 2.3. The invigilation staff scheduling won't be covered further in the thesis. For further reading about invigilation staff scheduling, Al-Yakoob et al. (2007) article is recommended. Similarly, a paper by Kassa and Tizazu (2013) is recommended where an integer programming model is proposed for an optimal weekly shift scheduling for a hotel but the shift scheduling is similar to the assignment of invigilation staff to shifts in the examination period.

2.2. Examination scheduling

One of the earliest published articles on examination scheduling can be found in Broder (1964) paper *Final Examination Scheduling*. The main purpose for his formulation was to eliminate mistakes that may arise when the scheduling is done manually and therefore achieve a minimal number of first order conflicts.

Cole (1964) similarly tried to minimize first order conflicts in his model for the examination scheduling by satisfying several conditions. In his approach, a sorting heuristic technique was proposed. A clash matrix was constructed for the courses, where a course was said to be incompatible with another if it had common students. Differently to Cole (1964), Broder (1964) used a random selection of assignments. By using such heuristic, course i was assigned to a period n and course ii was assigned to another period by considering the assignment of course i.

Wood (1968) solved the problem differently than Broder (1964) and Cole (1964). He extended the conditions on how one should construct an examination timetable by making the problem more realistic. Instead of just minimizing the first order conflicts as previously done, new conditions were added so that all students in the same subject had to be assigned to the same room. Secondly, that each room only holds a certain capacity of seats and lastly, that students facing two exams within 24 hours should be kept to a minimum. In occasions where that happens, afternoon and the morning after were selected in preference to same day examinations. In order to solve the timetable, a heuristics approach was proposed so that the examinations that needed the largest rooms were put in a descending conflict order. This process was then iterated for each room going from the largest to the smallest room. The courses were then assigned in this sequence to the slots such that no courses with common students would have a clash in the same slot. Using sequential heuristics as was done in his work was pretty common in early literature, but later it was replaced by constraint based techniques (Qu et al., 2009).

Johnson (1990) proposed a constraint based technique using integer programming to model the examination scheduling for the University of the South Pacific. The main goal in his work was to minimize consecutive examinations for students i.e. the number of students taking two exams the same day or afternoon/morning after. In the model proposed, four conditions were formulated such that no student would have an exam simultaneously, limitation of examination slots available and limited room capacity in each slot similarly to Wood 1968; Broder 1964; Cole 1964 but differently to their approaches, assignments of courses with the highest enrollments to the beginning of the examination period was done. By doing so, he allowed maximum time for marking. To solve the problem, a simple heuristic was proposed such that a feasible solution was found but was later improved by re-ordering heuristics.

Using a multi-phase formulation has also been common in the examination scheduling literature. Arani and Lotfi (1989) and Vahid Lotfi (1991) solved the problem using more than one phase. Arani and Lotfi (1989) used a three phase approach. The first phase assigns the exams into blocks such that the number of students facing one or more exams in a block was minimized. In the second phase, the assignment of exam blocks to exam periods took place using the solution of phase one as an input. The third phase was used to schedule the assignment of exam blocks to exam periods in each day such that two consecutive exams would be minimized. Vahid Lotfi (1991) used a four phase approach in his work. The first phase assigns the final exam to exam blocks where the objective was to minimize the total number of students facing two exams at the same time. Phase two assigns the exam blocks to exam days where the objective is to minimize students having two or more exams per day. The third phase arranges the exam days and exam blocks within days by minimizing consecutive examinations. In the fourth phase, the assignments of classrooms to exams took place where the objective was to maximize the space utilization.

In general, it can be said that the examination timetabling problem is a well researched problem and many methodologies have been used. As a result many comprehensive surveys have been performed on the literature. Schmidt and Ströhlein (1980) proposed a complete bibliography of the work in the early literature. Later, Carter (1986) published a survey over practical applications in universities. There, it can be seen that most of the work reviewed, the main goal was first and foremost to minimize consecutive examinations even though different conditions were formulated for each case. Although, that condition is critical for the problem, other factors are as vital for the quality of the solutions and for the students' comfort as well. Similarly, all of the work reviewed in their survey had only been implemented in one institution. A more recent survey can be found in Qu et al. (2009). In that survye, most of the methodologies used to solve the examination timetabling problems are presented. Other great surveys that are worth reading can be found in Schaerf (1999), Carter and Laporte (1996) and Burke and Petrovic (2002).

Carter et al. (1996) introduced 13 benchmark data sets which have been widely used in the examination scheduling literature for researchers to test their approaches on real data rather than randomly generated problems and to compare their results to others. From 2002, three international timetable competitions have taken place where the main goal was first and foremost to make a common ground for comparison of the wide variety of the methodologies proposed (McCollum et al., 2010). All three competitions have been different in nature. The main task of the first competition ITC2002 was to construct a course timetable by satisfying several conditions (Metaheuristics Network, 2002). In the ITC2007 three tracks were introduced. The first track was a post enrollment examination scheduling problem and the latter two tracks were correlated to the course timetabling (post enrollments and curriculum based timetabling). A solver proposed by Müller (2009) was successful in the competition by achieving the best results for two out of the three tracks (curriculum based course timetabling and the examination timetabling) but the algorithm proposed was based on several techniques. The third timetable competition ITC2011 focused on high school timetabling which has not been popular in scientific terms (Post et al., 2016). From what researchers submitted to those competitions, it is obvious that there is a high interest in competitions like that in order to compare methodologies to others. More benchmark data sets have been proposed but a good review on the data sets along with results on the benchmark problems can be found in Qu et al. (2009).

The examination timetabling problem is NP complete and therefore really difficult to solve (Cooper and Kingston, 1996). What makes timetabling challenging is the freedom of choices of the students (Laporte and Desroches, 1984; Amaral and Pais, 2016). Most of the articles that have been reviewed in this section use some sorting heuristics in order to achieve a solution since the problem is computationally heavy. Carter et al. (1996) compared five types of heuristics on real world problems and randomly generated problems. The methods investigated in his work are the largest

degree, largest weighted degree, saturation degree, largest enrollment and random ordering. The Saturation degree gave the best result in the lowest computing time. In a similar study, largest enrollment was shown to give the best result in a real world problem (Kahar and Kendall, 2010). Other forms of heuristics that have been used to solve the examination scheduling are so called meta heuristics and hyper heuristics. Using meta heuristics is more problem specific while latter method can be applied to any optimization problem since it usually chose the most suitable low level heuristics for the problem. (Bilgin et al., 2007; Pillay and Banzhaf, 2009; Burke et al., 2007). Other approaches have also been used to solve the examination scheduling such as tabu search, ant algorithms, evolutionary algorithms, simulated annealing and great deluge, graph coloring, constraint based techniques and clustering techniques among others, have also been used (Qu et al., 2009; Eley, 2007).

Some papers on real life applications can be found using integer programming (IP) to solve the examination scheduling. McCollum et al. (2012) proposed an extensive examination scheduling formulation. The work in that study was implemented in Europe, Australia and America with good results. Similarly, did Al-Yakoob et al. (2007) propose IP models for the examination timetabling which included some specific conditions that were solved such as gender related policies and traffic consideration along with other regular conditions for Kuwait University. The model was formulated in a two phase approach where the first phase assigns exams to time slots and rooms and the in second phase assignments of invigilation staff takes place. Dimopoulou and Miliotis (2001) did similarly use IP formulation in their work of implementing both course and examination timetabling system for Athens University of Economics and business. In order to construct the examination timetabling, the course timetabling was used to generate an initial infeasible solution for the examination scheduling since it had no conflicts. Later it was improved by a heuristic approach.

Not all of the methodologies proposed solve the examination scheduling in the same manner. Most of the work reviewed in the last paragraphs propose a post enrollment scheduling where the student enrollments to the courses are known beforehand (Lewis et al., 2007). However, some universities do schedule without knowing the number of students that will be enrolled in the course. Such problem have mainly been solved for the course timetabling before and not for the examination scheduling until recently(Müller and Rudová, 2016; Bettinelli et al., 2015; Bonutti et al., 2012). Cataldo et al. (2016) proposed a curriculum based examination timetabling with a three phase model using MIP along with one initial data processing phase to solve the problem for Universidad Diego Portales. In the initial phase the courses are clustered in groups, the second phase schedules the clusters to time slots. In the third phase the courses within each cluster are scheduled and lastly, rooms are assigned to the courses.

2.3. Room scheduling

Most of the work in examination timetabling literature only schedule the examinations to time slots such that the total number of students assigned to each time slot does not violate the seats available but not to rooms. A possible reason for this is the lack of extensive data and the fact that the problem can somehow be harder to solve (Kahar and Kendall, 2010). Although a feasible solution for the examination scheduling is achieved, that solution does not necessarily guarantee a feasible solution in the room scheduling under some circumstances (Dammak et al., 2006).

Laporte and Desroches (1984) solved the classroom assignment in their model for the examination timetabling. Firstly, it was presented that the room availability in each time slot can differ. Secondly, each course may only take place in a certain room type and finally, a room allocation heuristic was proposed. The heuristic ordered the examinations and the rooms to a decreased order then assigned the biggest exams to the biggest rooms. When a room was full but not all of the students in the course were assigned, a new "course" was made with the remaining students. Similarly, if a room was not full a new room was made with the capacity that was left of the original room. However, the remaining capacity and the rest of the students were then moved to appropriate ranks in their list. This process was repeated until all exams were assigned to rooms. No limitation is on how many exams were assigned to each room but in the final step of the heuristic, some balancing of students between rooms takes place.

Leong and Yeong (1990) proposed a model to assign rooms to exams after they had been scheduled to time slots. The main goal was to minimize the rooms being used and to assign the courses who need a particular room. Additionally, three constraints were implemented such that the room capacities could not be exceeded, the maximum number of exams in a room is limited to a certain number and an exam should preferably be held in one room if not it should be assigned to the nearest cluster of rooms. A heuristic approach was proposed such that it firstly assigned the courses with the special room needs to the appropriate room/s. Secondly, the remaining capacity of all rooms is calculated and the course with the highest enrollment is assigned to the room with the highest remaining capacity. This method is similar to Laporte and Desroches (1984) and provided a good room efficiency. The main difference between the methods is that when a course needs more than one room it is assigned to a neighborhood cluster of rooms.

Vahid Lotfi (1991) proposed an IP model and a heuristic procedure for the room assignments for the State University of New York at Buffalo (SUNYAB) room assignments. The model used a similar approach as Laporte and Desroches (1984) but instead of ordering the courses and rooms to a decreased order, the first exam on the list was put in the first room of the list with enough capacity, after the second exam

was assigned to the next room available on the list until all of the examinations were assigned to a room. The main drawback of the heuristic which was originally used in this case was when courses had to be split between rooms. When that happened, they often ended up on different campuses which were inconvenient for the students and the teachers. Therefore, a non-linear model was proposed in his work such that if an exam had to be split, it would be split within a campus.

Dammak et al. (2006) proposed an IP model of the room assignment for the examination timetabling. Similarly, a simple heuristic was developed to reduce the size of the problem. The heuristic tries to find a solution where only one exam is assigned per classroom. When that fails, the constraint presented in the integer programming is relaxed and will try to find a solution with two exams per classroom. The heuristic technique used consisted of 11 steps and 4 propositions which are in some manner similar to the technique of ordering technique as previously proposed by Laporte and Desroches (1984); Vahid Lotfi (1991); Leong and Yeong (1990).

Kahar and Kendall (2010) proposed a model for the room assignment for the examination timetabling and new constraints that had not at that time been modelled before in the literature. Those new constraints included formulations of distance between examination rooms and splitting examinations to multiple rooms. Similarly, a constructive heuristics was proposed.

Müller (2016) proposed similar constraints to his model. There, a stepwise penalization with different costs was used for a room split of each examination, getting higher as each examination was split into more rooms. However, an upper bound was put on how many rooms each exam could be assigned to if it was split. An implementation of a room split distance constraint was also done. There, a distance matrix between rooms in meters was introduced and as the assignments of a split examination into to rooms was further away, the more penalty it received. A new approach was also discussed in his work that would allow some examinations to be split into multiple periods in order to decrease the number of conflicts. Being able to split an examination into multiple periods seemed interesting by the author but questions arose if a such constraint is fair for students although improvements could be achieved.

2.4. Commonly used constraints

In this section, hard and soft constraints will be given a short introduction. Similarly, the most commonly used constraints for the examination scheduling in literature will be given special attention.

In general, when talking about constraints they can either be hard or soft. Hard constraints are in general constraints that can not be violated, e.g. some courses require exams and those that do must be allocated in the examination scheduling in one slot. Soft constraints are not as essential for problems but good to have when modelling. This can be constraints such as preferring some buildings where the examinations take place and such cases can be violated if it can not be satisfied.

Burke et al. (1996) proposed a survey over the examination timetabling in British Universities. The survey was sent to 95 universities which 56 replied. The main purpose was to see if the problem was similar in those universities in order to see if some method or algorithm could be designed that could produce timetables for all the problems. As a result, 13 common constraints were listed with additional 19 constraints that are also being used. Hence, it is easy to see that the problem varies between universities and even within each one. Consequently, a method or an algorithm that solves all problems for all universities is not possible. In Table 2.1 the 13 most typical constraints proposed by Burke et al. (1996) are listed. Although the constraints in Table 2.1 are the most common ones, many other constraints are needed for each case. Each school differs from another regarding the requirements meaning that producing a general solution for all schools is difficult Kahar and Kendall (2010).

	Constraint
1	There must not be more students scheduled to a room than there are seats.
2	Exams with questions in common must be scheduled in the same period.
3	Some exams may only be scheduled within a particular set of periods.
4	Only exams of the same length may be scheduled in the same room.
5	Exams with the most students must be scheduled early in the timetable.
6	Some exams must only take place in particular rooms.
7	Large exams halls must be scheduled in preference to smaller ones.
8	Exam A must be scheduled before exam B.
9	No student is scheduled to exams in two consecutive periods.
10	No student is scheduled to more than one exam in any particular day.
11	Each student's exams must be evenly spread throughout the timetable.
12	No student is scheduled to exams in two consecutive days.
13	Exams must be scheduled to rooms near to the relevant department.

Table 2.1: The most commonly used constraints

2.5. Summary

In this chapter, a brief overview on educational timetabling has been given. In general, it can be classified into two sub-problems or to the *Course timetabling* and *examination*. The examination timetabling is then distinguished to three sub-categories or the *Invigilation staff scheduling*, *Examination scheduling* and *Room scheduling*. The only sub-problems that will be modelled in this thesis are the examination scheduling and the room scheduling. Therefore, the invigilation staff scheduling was not covered further. A literature review was then given on work on the examination scheduling and room scheduling respectively and from there it is obvious that many methodologies have been proposed. Lastly, the 13 most commonly used constraints in the examination timetabling problem were presented.

3. Examination timetabling at the University of Iceland

In this chapter, the examination timetabling problem at the University of Iceland will be described in detail and the problem will be addressed. Section 3.1 will present the problem description at the University of Iceland and Section 3.2 will present the model design and list all of the key conditions that must be met when making the models. Section 3.3 will present all of the data that will be acquired for the models but it will be acquired manually and from UGLA. Lastly, Section 3.4 will present the codes and transformation of the data with a brief functionality description.

3.1. Problem description

University of Iceland (U.I) is one of the biggest organizations in Iceland with 1.577 employees on a permanent base and around 2.309 part time teachers yearly. The university consists of five schools which are divided into multiple departments. There are many programs within each department. Each program has at least one field and each field consists of one curriculum. Usually, each student is just enrolled in one program and one field. However by exception, are some students registered in more than one program or field. Today, about 13.200 students are enrolled to U.I as can be seen in Figure 3.1 (University Of Iceland, 2017, 2016).

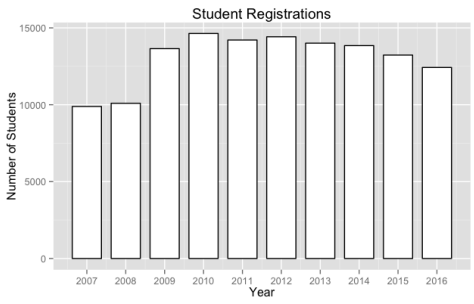

Figure 3.1: Student registrations at the University of Iceland

Most of the students enrolled at the U.I are assigned to a course that requires them to pass a final exam in order to pass the course. In general, the final exams can either be *written, verbal, practical* or *computer based*. No matter what kind of an exam it is, it must be scheduled to time slots in the examination period.

Each exam has multiple resources but Figure 3.2 shows the main ones. In order for an exam to take place, it must satisfy all of its resources. Each exam must have students enrolled in the course and a teacher. If so, the exam can be scheduled to a time slot and secondly assigned to building/s, room/s and seats. Lastly, invigilation staff must be assigned to shifts in each room where an exam takes place.

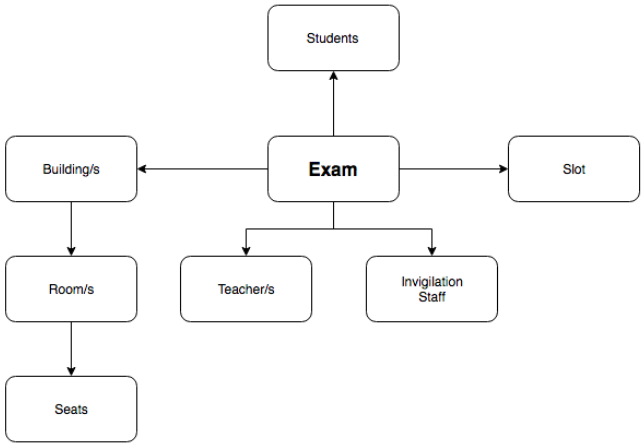

Figure 3.2: Resources for each exam

Past years, all of the examination timetabling has been done manually by the director of examinations, requiring about 5 to 7 full work days for each main examination period which are held at the end of each semester. Each main examination period requires 11 to 14 days, depending on how many exams and students are supposed to be assigned. Typically, two 3 hour time slots are available per day, one from 09:00-12:00 while the other is 13:30-16:30. The majority of the exams require 3 hours while some of the exams may require a longer examination time or even shorter.

There are two shorter examination periods that follow the main ones for the students who had a valid reason for their absence. Those two exam periods are usually just 5 days respectively for each semester. Hence, examination timetabling takes place at least 4 to 5 times per year.

As the years go by, the process is always getting more computer based than it was before. Today, the examination director uses a system that was originally built by Reiknistofnun (*the operator of the University's IT systems*) to make the timetabling easier than it used to be. However, this system which is operated through UGLA(*inner network for students and teacher*) is not based on any algorithms or models to help the scheduling process itself in a direct way. The only

thing the system does, is to visualize clashes between students in the same time slot, show the distribution of students over the examination period for a single course and the total number of students assigned to each slot as can be seen on Figure 3.3. The figure shows an example of a course with 321 students enrolled. That course is assigned to a dummy slot (23.12.2016) and is indicated by a red color. The examination assignments of the students of that exam to other exams can be seen and is indicated by a green color. Therefore, the only place were this exam can be assigned is on the dates 12-16th of December to avoid clashes and to give the maximum period spread. In the dummy slot, 4439 examinations are left to schedule.

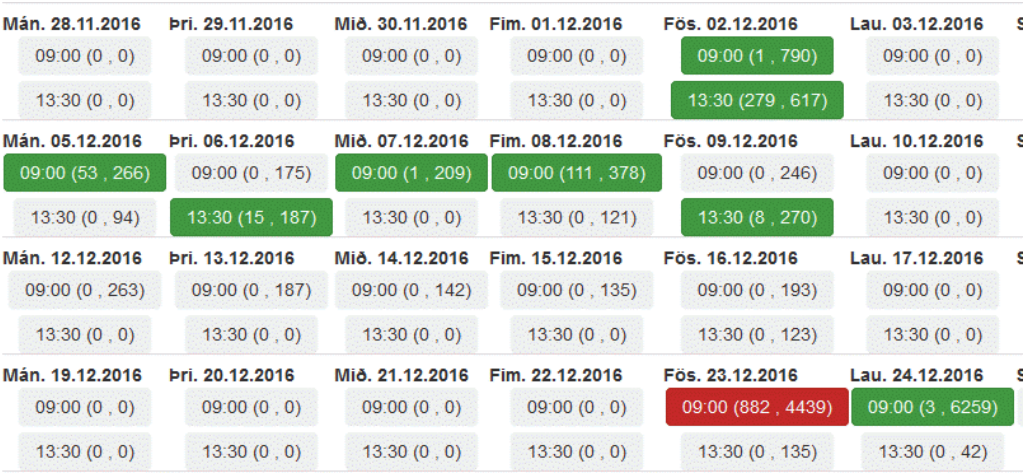

Figure 3.3: Snap shot of UGLA examination scheduling visualizer

The director of examinations has years of experience in making the timetabling manually. He has developed some guidelines which he follows while making the timetable. Firstly, it is quite common that courses require being assigned in the beginning of the period while others require specific time slots. Courses that require being assigned in the beginning of the timetable usually have a lot of international students enrolled. This is usually required for the autumn timetable to allow the international students to finish as early as possible in order to get back home early for Christmas. Similarly, due to dark December mornings in Iceland, the courses that have international students are usually assigned to the second time slot of the day rather than the first, if possible.

In general, a student should not be facing two examinations in the same time slot. Similarly, the assignments of students to two exams during the same day and within 24 hours (*i.e. exams in the afternoon and the morning after*) should be kept at a minimum and preferably, should students receive at least one day off before every exam. However, by exception, this cannot be satisfied for every student since they are permitted to choose subjects within a department and between schools. Similarly, the time slots have a limited amount of seats and therefore it can be difficult to satisfy restrictions of the same day examinations and examinations within 24 hours.

However, what makes the room scheduling difficult, is to have two or more large courses assigned to the same time slot due to limitations of large rooms. If more are than one is assigned, the courses must be split among many buildings. Therefore, the rule of thumb is to only assign as few large courses as possible in a time slot. Preferably, the number of students assigned to each time slot should be as equal as possible over the examination period.

Other guidelines are connected to the room and building assignments. In general, the rule is to have two courses assigned to the same room in order to avoid cheating by students. At the same time should exams with different duration not be assigned to the same room in order to keep the disruption at the minimum. Disruption is not only bound to students finishing at different times but also to teachers visiting each exam. By request, teachers visit their exams two times in a three hour examination time. Therefore, having too many courses assigned in a small room can inconvenient for students. Hence, the rule of thumb is to assign up to three courses to a room if the capacity of the room is greater than 20 seats but otherwise two. Similarly, to split a course between rooms and buildings should be kept at a minimum. Each course should be divided into equally big groups such that the minimum number of students from a course are always assigned to a room. However, if the students are 12 or less in the course, they should only be assigned to one room.

Each school and department have their own set of buildings where the courses are usually taught. Consequently, courses within each department have their own preferred building to be assigned to. If the courses cannot be assigned there, they are assigned to their required buildings if possible. The required buildings are buildings that are located close by their preferred building. When that fails, they will be assigned elsewhere at the central university area. This is done in order to have the students familiar with the rooms and the location of them. Exams are however required to be assigned to specific rooms outside their preferred and required buildings if they require a computer room. In general the rooms available can be classified into *general rooms*,*computer rooms*, *special rooms* and *special computer rooms*. Consequently, each course must be assigned to the right type of room for each exam. Whether the exam will be a general one, computer based or requires to be assigned to special rooms it must be satisfied for all examinations. However, not all exams are supposed to be assigned into rooms even if they are being scheduled into the examination table. These exams are particularly verbal exams or other exams such as home-exams. Other requirements are regarding courses with questions in common but they are usually assigned to the same set of buildings so that the teacher does not need to go to several buildings. Similarly, the exams that require computers should be spread over the examination period as equally as possible.

Many requests arrive through emails or in other ways to the director of examinations. The requests that arrive through emails can be really diverse. Exams that must be examined together (*courses with questions in common*) often arrive through email

but not all the time. Similarly, the departments and schools may require exams to be in a specific week or slots (*i.e. the beginning of the examination period or at the end*). Some schools even have their suggested examination schedules. Similarly, in a few cases, teachers may require specific times lots due to their availability over the examination period.

Students in some courses may have extended examination time due to dyslexia or other disabilities. Usually, the extension time is extra 15 minutes per hour but can be longer. These students are not assigned to the same set of rooms as the others and do not follow the same assignment rules. Up to six courses can be assigned to each room in those cases. These students often require multiple things. Some may require a seat next to a window, some near the exit, being alone in the room, some require a computer while others require colored exams. However, this list is incomplete but sufficient for most of the students with disabilities. Similarly, there is a a limitation on how many students with disabilities can be assigned in each slot.

In general, the total number of rooms assigned to a course should be as few as possible. Similarly, when a course is split into multiple rooms it should preferably be assigned in one building, if not it should be assigned to rooms within a cluster of buildings but simultaneously to as few buildings as possible. Three main clusters are defined at the university area: *Melurinn*, *Torfan* and *Holtid*. If a course cannot be assigned within those clusters, which consist of multiple buildings, it should only be assigned to one building. Similarly, the total number of rooms assigned to each time slot should be as few as possible due to the limitations of invigilation staff and to open a new room means more cost for the university.

3.2. Model design

In the last section, an overview was given about U.I and the students and the guidelines which are used while constructing the examination timetable. In this section the key conditions that must be taken into consideration when designing the models will be listed, their objectives and its design will be discussed.

The models will be designed in a two phase approach. Phase I will be used to assign exams to time slot while Phase II will use the results from Phase I as an input and schedule exams to rooms. This two phase method is chosen since the timetabling is performed in that way by the director of examinations. The main goal for Phase I is to schedule the exams in a such way that it minimizes the clashes, the students having examinations in the same day, students having examinations within 24 hours and students not receiving a day off before an exam. At the same time, the computer exams should be spread over the examination period equally and

	Key Conditions
Phase I	Fix a course in a time slot
	Assign a course to only one time slot
	No Student Clash
	Courses with questions in common must be assigned at the same time slot
	Not two exams in a day for a student
	Not two exams within 24 hours i.e. afternoon and morning after
	Have a day off without examination
	Maximum number of students assigned to each time slot
	Just one large course in a time slot
	As equal spread of computer based examinations
	As equal assignments to each time slot
Phase II	Assign all students to seats
	Assign special students with disabilities to special rooms
	Do not violate room capacity
	Exams with different durations should not be assigned to the same room
	Not too many rooms if a course is split into multiple rooms
	No split if a course has 12 or less students
	Not too many courses in each room
	As few buildings as possible
	Assign exams to their preferred or required buildings
	Assign a course to required room
	No assignment to specific rooms

Table 3.1: Key conditions for the model design

total number of students assigned to each time slot should be as equal as possible over the examination period. Phase II will schedule the exams to rooms in a such way that the total number of rooms and buildings assigned to each exam will be minimized. At the same time, will the exams be assigned to their preferred and required buildings as often as possible.

In Table 3.1, all of the conditions that must be formulated are listed for the models in Phase I and Phase II. By looking at Table 2.1 in Section 2.4 it can be seen that 10 out of the 13 most common constraints are being used at U.I but for the problem addressed in this thesis, there are 21 key conditions that must be satisfied. This seems to confirm that each university differs from each other regarding requirements for timetabling. In the last paragraphs, the key conditions were discussed for the models. All of these conditions have been discussed in more details in Section 3.1 where the problem description was given for the U.I.

3.3. Data

In this section, the data that will be acquired for the model will be described briefly. The examination timetabling will be solved in a two phase approach as has previously been discussed in Section 3.2. Therefore, all of the data regarding the exams and the students will be acquired for Phase I whereas all data regarding the rooms and their assignments will be acquired for Phase II. The data that will be used in this study for the University of Iceland will be acquired in two ways. Firstly, it will be acquired manually and secondly from UGLA. UGLA is an inner web of U.I for students and teachers and therefore contains all of the courses and their enrollments. The data acquired from UGLA is really dynamic, meaning it changes on a daily basis. Hence, all of the data that will be used in this thesis is the data that was available 4 days before the publication of the official examination scheduling itself generated by the model found in this thesis. In the next two sections, a brief description will be given on the data that was acquired.

3.3.1. Manual collection

A lot of the courses have some sort of requirements regarding their assignments to time slots but these requirements might be requests from teachers teaching the courses, the departments or the director of examinations. This information usually varies from year to year and must be collected manually. A lot of the requirements arrive by email to the director of examinations. He needs to keep track of every one of them and check if it is a reliable. When a request gets approved, it is entered straightly to the data used for the model along with other accepted requirements. At the same time, information about courses with questions in common is collected manually but such courses must be assigned to the same time slot. As before, this information often arrives through emails but some of it can be found on UGLA. This information is similarly entered straight into the data used for the models by the director of examinations. However, by entering this into the data, he must transform all of the Icelandic letters to English alphabetic letters. In this thesis, this step is performed in that manner since it was the simplest way of making this data. However, later this step will be done differently and perhaps through a graphical interface.

Another data that must be acquired manually is regarding the building/s that the courses should preferably be assigned to. All departments within each school have preferred buildings where their examinations should be assigned if possible in order to have the students familiar with the housing. In Table A.1 in Appendix A all of the preferred buildings are listed for each department which should be the number one choice to assign the examinations for each department to. However, this is

not always possible. Consequently, another list of buildings which are used as a second option (called required buildings) whenever an exam can not be assigned to its preferred building. The required buildings are usually a list of all the buildings at the main university central area with slight variations between departments. In Table A.2 in Appendix A all of the required buildings are listed for each department. All of the description of the codes needed for the transformation to the right format with a brief description of the flow can be found in Appendix B.2.1.

The last data that was acquired manually was made by the director of examinations which was an Excel file with all of the rooms and buildings and their elements. In that file, all of the rooms and which building they belonged to is stated. Similarly, the rooms are categorised if they are used for general examinations or for students with disabilities and whether they are computer rooms or not. Other items such as the room capacity and a coefficient from 1 to 3 are given for each room. If a room receives the coefficient 1 it is preferred for examinations while a room with coefficient $\mu = 3$ is not really suitable. The transformation of the data to the right format and the flow of the code used for the transformation can be found in Appendix B.1.7.

3.3.2. Data from Ugla

Most of the data used in this study are acquired from UGLA which returns the data as Excel files. Three files were acquired which are all used for different purposes. The first file contains information about the student registrations to all courses of U.I. In that file, each line represents each registration by a student to a course by his ID number and which department he belongs to. Not all of the courses found in this file require a final examination and therefore, they are filtered away to have the right set of courses. That step can be found in Appendix B.1.1 with a brief description.

The second file acquired from UGLA consist of the courses that should be examined along with their department and school number. In the file, each course receives a short name and a longer name. Similarly, the total number of students assigned to each course can be found along with which type of the examination each course requires. This data is mainly used to construct a list of the required buildings and the most preferred buildings for each course since this file has the department number for each course. That step and a brief description can be found in Appendix B.1.4.

The last file acquired from UGLA is an Excel file regarding all of the courses that should be examined but in total 525 exams should be examined. At U.I, each course receives three names: one short name (*ENS101G*), long numerical identification (05150220166) and a long official name (*Eðli tungumálsins I: Hljóð og orð*). The

long identification must be used to import the solution to UGLA. Other elements that can be found in this file are the type of examinations and the duration of each exam. Not all of the examinations require to be assigned to a room during the examinations (*usually verbal and home exams*) while other require a computer for their examinations. Therefore, lists can be constructed on which examinations must be assigned to rooms and what kind of rooms. The total enrollments to each course can be found along with the total number of students with disabilities for each exam. Hence, from this file, the count of the special students will be used. The transformation of the data to the right format and the code used for it can be found in Appendix B.1.3.

3.4. Functionality

In this section, an overall map of the system will be drawn. The system itself which consist of the models, multiple codes and transformations of the data as has been previously described in Sections 3.3.1 and 3.3.2. This step is considered to be essential so that the reader can get a better understanding of how the system works. The system is divided into two individual functional parts. Part I, will cover all codes and transformations of the data for Phase I (*the examination scheduling*) while Part II will cover all codes and transformations of the data for Phase II (*the room scheduling*). The sections will only present two overall system maps along with brief descriptions. A more detailed description of each step in the system maps can be found in Appendix B.2.1 and B.2.2 for Parts I and II respectively.

3.4.1. Part I

The main objective of this section is to present an overall system map for Part I of the system, which consists of the model, all codes and transformations of the data for Phase I which represents the examination scheduling itself. In Figure 3.4 can the system map for Part I can be seen. The system consists of multiple steps where each step has been numbered. Firstly the main data (*marked* $1, 2, 3, 4$) along with the model in part are converted to an *lp* file using GLPK and the solver GLPSol. An *lp* file consist of all possible combinations of the constraints and the objective from the data supplied. After that step has been performed, the *lp* file is used as an input for the optimization process (7) using Gurobi as a solver. When the optimization has finished the right information needed for the next step is filtered and used for further processing. Steps 9 and 10 are similar but the main difference is the transformation of the data. In step 9, the data is transformed in a certain way in order to use it as an input for Phase II. After, the solution has been transformed in step 10 it is

used as an new data for part 11 which works in a similar way to part 8. The only difference, is the new data supplied in 10 as an input and that the GLPSol will be used to print the solution to the right format that can be imported to UGLA. For a more detailed overview of each numbered step appendix B.2.1 is recommended.

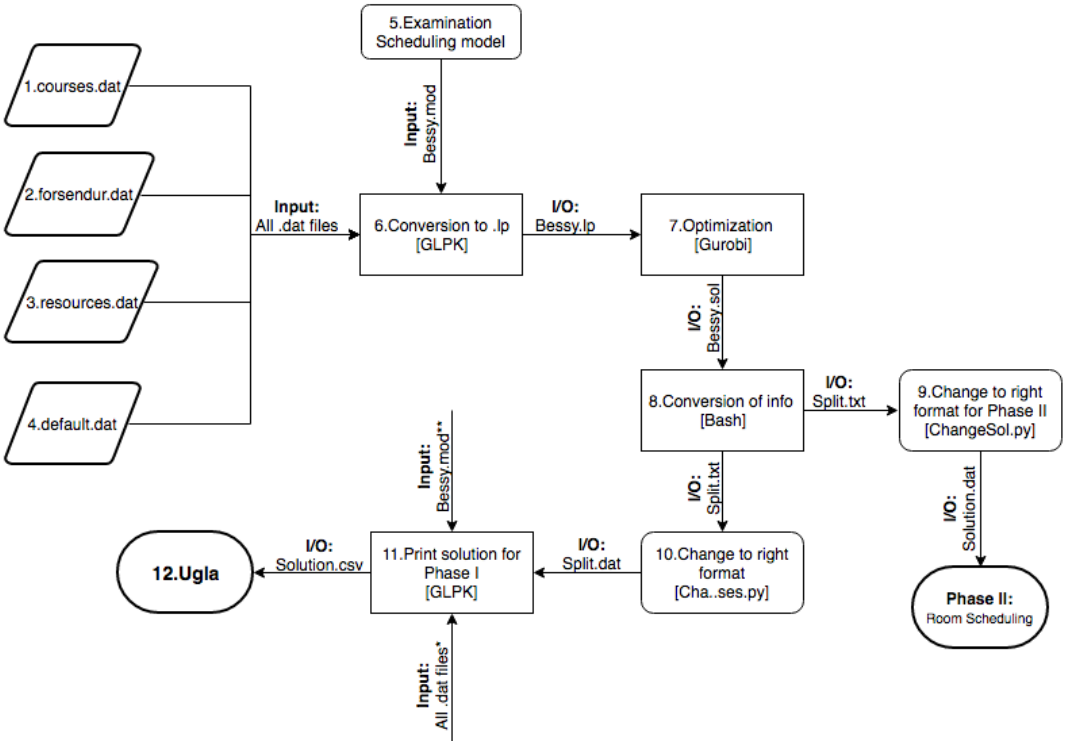

Figure 3.4: The system map for Part I which represents Phase I and all of its parts[1]

3.4.2. Part II

The main objective of this section is to present a system map of Part II of the system, which consists of the model, all codes and transformations of the data regarding Phase II, which represents the room scheduling. The overall system map can be seen in Figure B.17 where each part of it has been numbered. This part follows the same principles as in Section 3.4.1 but here, some new data and a model are presented. The connection between Parts I and II can be found in the data in part 8 where the data with the solution from Phase I is imported. All of the data (parts $1, 2..9$) and the room scheduling model (10) is converted to an *lp* file using GLPK and the GLPSol. After the *lp* file has been made, it is used as an input for the optimization process using Gurobi as a solver. After the solution is ready it is written in a new file and transformed to the right format as done in step 13. In step

[1]The * indicator that can be found on the figure corresponds to the data marked by the numbers $1, 2, 3$ and 4.

14, the GLPSol is used to print the solution to the right format that can be used to import the solution to UGLA. In this step, all of the data previously used for step 11 and the model are used again along with the new data that contains the solution as transformed in step 13. Since each slot is solved individually, the data in part 9 must be changed every single time for each run. For further details on each step on the map, Appendix B.2.2 is recommended.

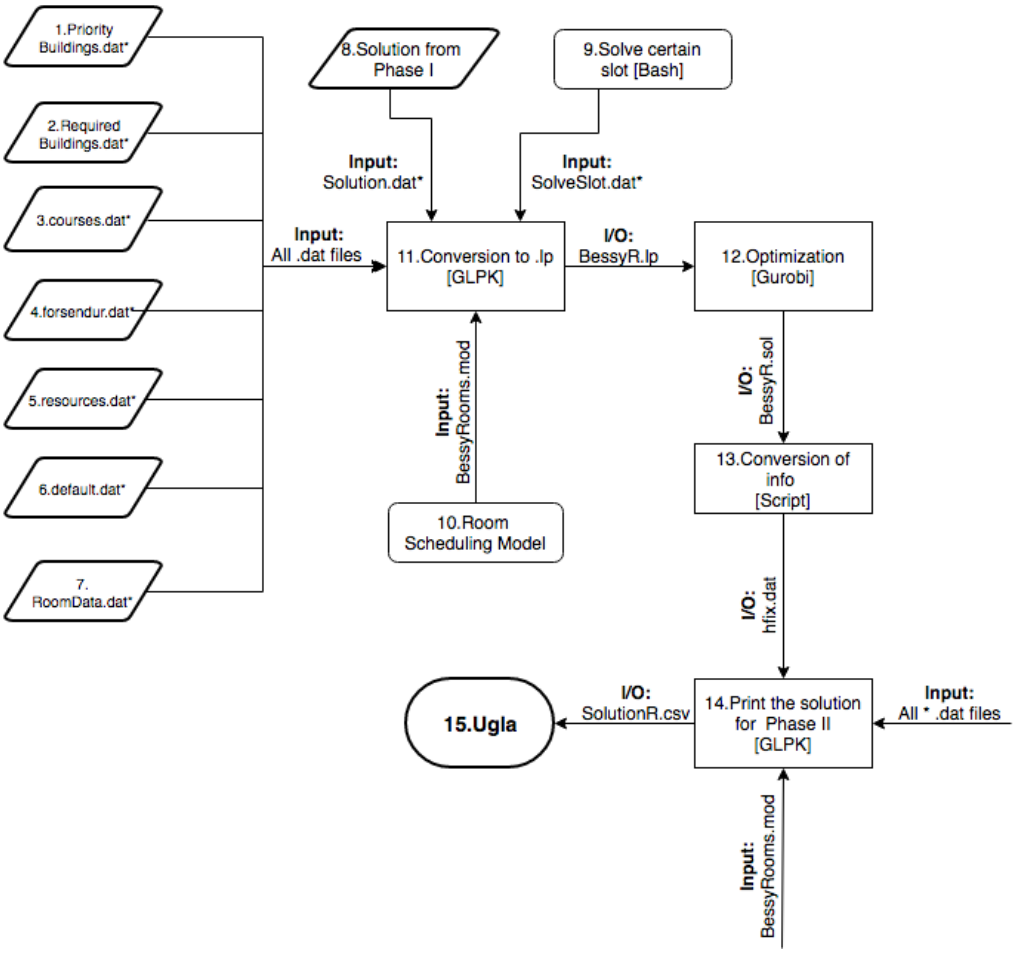

Figure 3.5: The schema for the room scheduling and all of its parts

3.5. Summary

In this section, the problem description has been discussed for the examination scheduling at the University of Iceland. As the situation is now, the examination scheduling and the room scheduling is performed manually taking $5-7$ days in general for each semester excluding the time needed for the additional examination periods which are usually after the main ones for makeup exams. After the problem

description had been discussed the model design took place where it was decided to split the problem into two phases: *Phase I* and *Phase II* since the problem has been solved in that way by the director of examinations. Phase I of the model will therefore represent the examination scheduling where the courses will be assigned to suitable time slots in a such way that no clash will occur, the number of students having same day examinations, examinations within 24 hours and not receiving one day off before an exam will be minimized by meeting various requirements. Similarly, the computer based examinations should be spread as equally as possible along with the total number of students assigned in each slot. Phase II will represent the room scheduling where the exams will be scheduled into rooms. The total number of rooms and buildings assigned to an exam will be minimized by meeting various requirements. Similarly, will the exams be assigned as often as possible to their preferred buildings but if it cannot be satisfied they are preferably assigned to their required buildings. The constraints that needed to be for Phase I and Phase II formulated were presented in Table 3.1. All of the data needed for the models was similarly presented in Section 3.3, but it will mainly be collected manually and from UGLA (inner net for U.I). Lastly, in Section 3.4 the overall system maps were presented with brief descriptions. In the following chapter, the data collected will be inspected and analyzed.

4. Data analysis

In this chapter, the data that was collected for the model in Chapter 3 will be inspected and analyzed. In Section 4.1, the data regarding the courses and students registrations will be analyzed and visualized in order to get a better understanding of it. Similarly, the network of the examinations will be plotted. In Section 4.2, all of the data regarding the buildings and rooms available will be inspected and visualized. Lastly, clusters of buildings at the university area will be defined.

4.1. Course data

In this section, the data regarding the courses and students registrations that was collected in Chapter 3 will be visualized and inspected. The data consist of 524 courses with 25295 examinations. Behind these 25295 examinations are 8901 students. In Figure 4.1, the number of examinations per student can be seen. The variation of the total number of exams per student can be from 1-8 exams. The majority of the students are assigned to 1-4 exams but only small percentage of the students go to 5 or more.

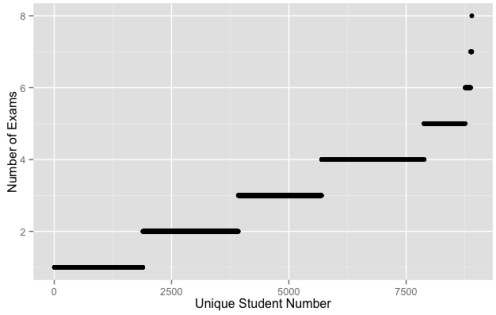

Figure 4.1: Numbers of examinations by students

In Figure 4.2, the courses are plotted in an increasing order by the total number of students enrolled in the courses. Most of the courses have less than 100 students enrolled while the largest course had 375 students. This can potentially mean that the assignments of exams to rooms should be easy for the vast majority of the courses due to their small sizes. However, the assignment of the larger courses can lead to difficulties where splitting of the courses between buildings and rooms will be unavoidable.

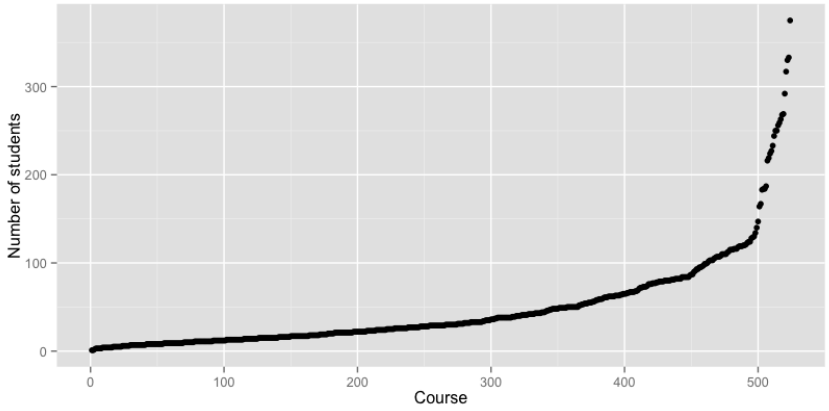

Figure 4.2: Total number of students assigned to courses in an increasing order

At the U.I there are five schools as can be seen in Table 4.1. It can also be seen that most of the schools have a similar total number of exams. However, School of Education consists of only 45 exams while other schools have the total number of exams between $116 - 124$. This is interesting to inspect more closely since the room allocation will need to assign courses to suitable buildings and rooms.

The largest school is the School of Social Sciences with roughly 7656 examinations of 122 exams. School of Engineering and Natural Sciences is the second in size with 6648 examinations in 124 exams as can be seen in Table 4.1. Therefore, it is most likely that the largest courses in the university belong to these two schools. This can mean that it will be difficult to assign two large courses from both schools in the same time slot due to limited availability of large rooms and there can possibly be a correlation between the large courses. The total number of exams at the School of Education is 45 and consist of 2428 examinations which makes it the smallest school regarding the examination. The total availability of seats at Stakkahlíð is 7854 over the 22 slots. Therefore, it is can be seen that all examinations of the School of Education can be assigned to Stakkahlíð (home of the School of Education) allowing more examinations in other departments to be assigned there if needed. More inspection of the buildings and rooms will be performed in Section 4.2.

Schools	Number of examinations	Number of Exams
Social Sciences	7656	122
Health Sciences	5483	117
Humanities	3140	116
Education	2428	45
Engineering and Natural Sciences	6648	124

Table 4.1: Number of examinations and exams per school

In Figure 4.3, the connections between all of the courses in examination timetabling are plotted. On the edge of the circle each individual course is plotted by size and colored by its department. If there is a connection between two courses, a line is drawn which represents that there is common student/s. As thicker the line is, more common students there are. From the figure, it is clear that the examination scheduling is a hard problem to solve since most of the courses are connected in one way or in another although belonging to different departments or schools.

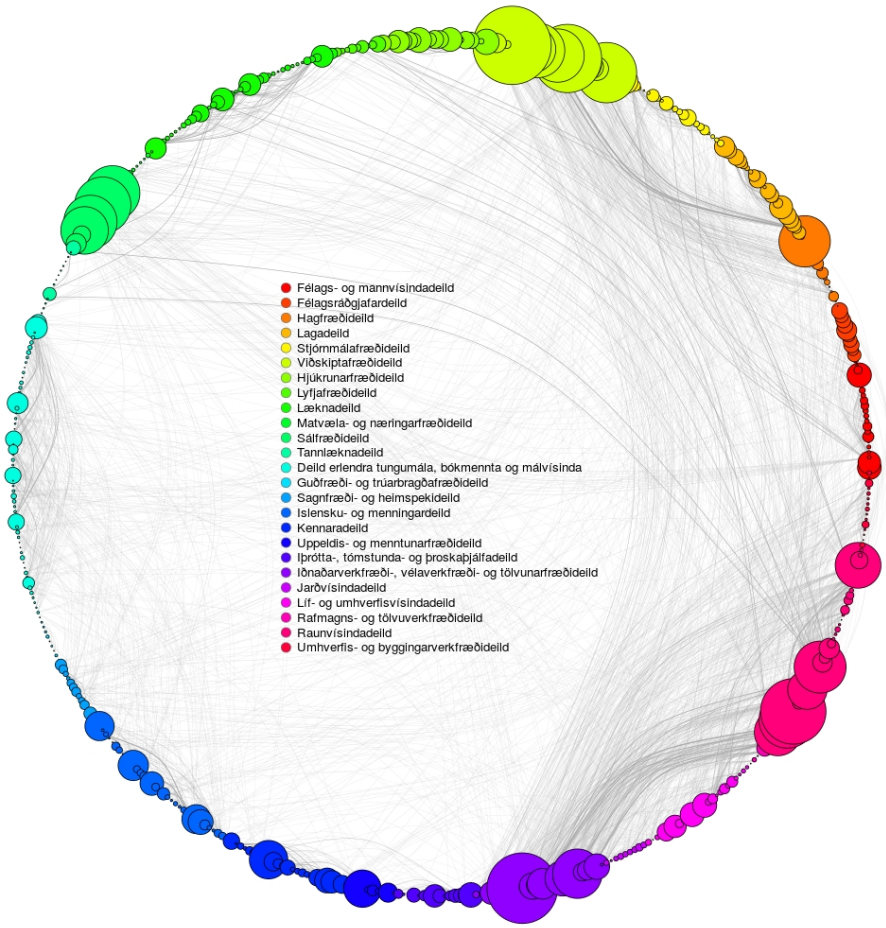

Figure 4.3: The connections of the examinations

4.2. Building and Rooms

In this section the data regarding the buildings and the rooms available will be given a closer look. At the U.I, there are 14 buildings suitable for examinations. In general, their location can be divided into the Central University Area, Stakkahlíð and Laugarvatn. Since Laugarvatn is located far away, the examinations should not be assigned there except for the courses taught there.

Figure 4.4 shows the total number of seats available in each building. In general, the rooms can be classified as *general rooms* which are rooms for general students,

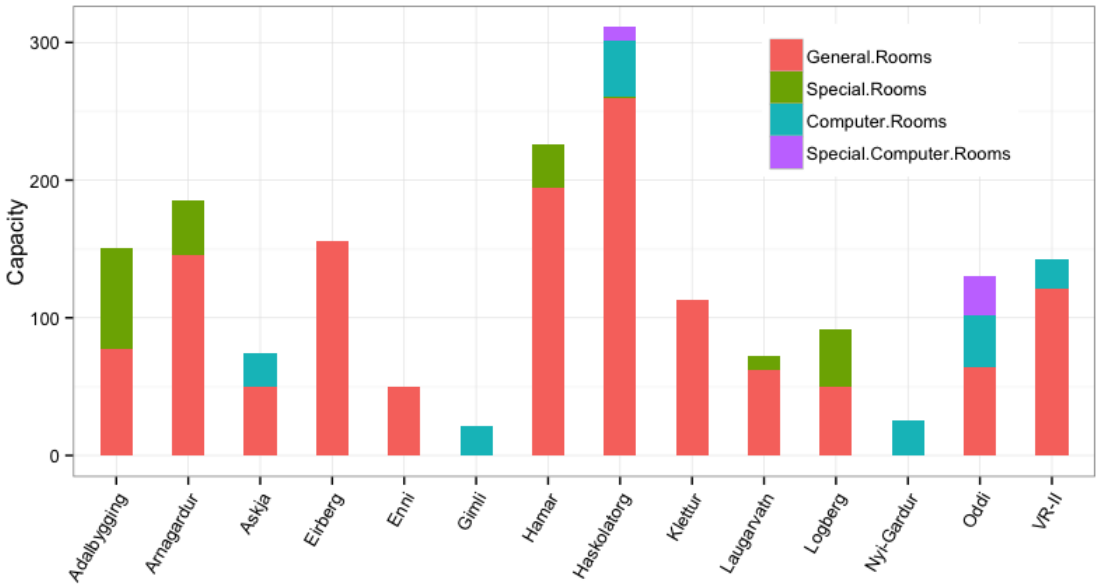

Figure 4.4: The capacity of the buildings and seat types

special rooms which are seats available to students with disabilities, *computer rooms* which are available seats for exams that require computers and *special computer rooms* which are rooms for students with disabilities that require a computer for their examinations. Table 4.2 shows the total number of seats available for each seating type at the U.I.

Seat Type	
General Seats	1343
Special Seats	198
Computer Seats	170
Special Computer Rooms	38

Table 4.2: Seating capacity by seat type

Not all of the seats available are suitable for examinations. Therefore, each room receives a priority coefficients μ in the data, which is defined on the interval $\{1, 2, 3\}$ depending on how much priority is on each room. When $\mu = 1$ the room is considered to be a suitable room for examinations while rooms with $\mu = 3$ are not considered suitable. Figure 4.5 shows the number of rooms in an increasing order by capacity and each priority order by colors. In total of 69 rooms are available where 38 rooms are classified as suitable rooms for examinations, 21 rooms have a medium priority. The last 10 rooms are not considered to be a really good choice for the examinations and should not be used at all.

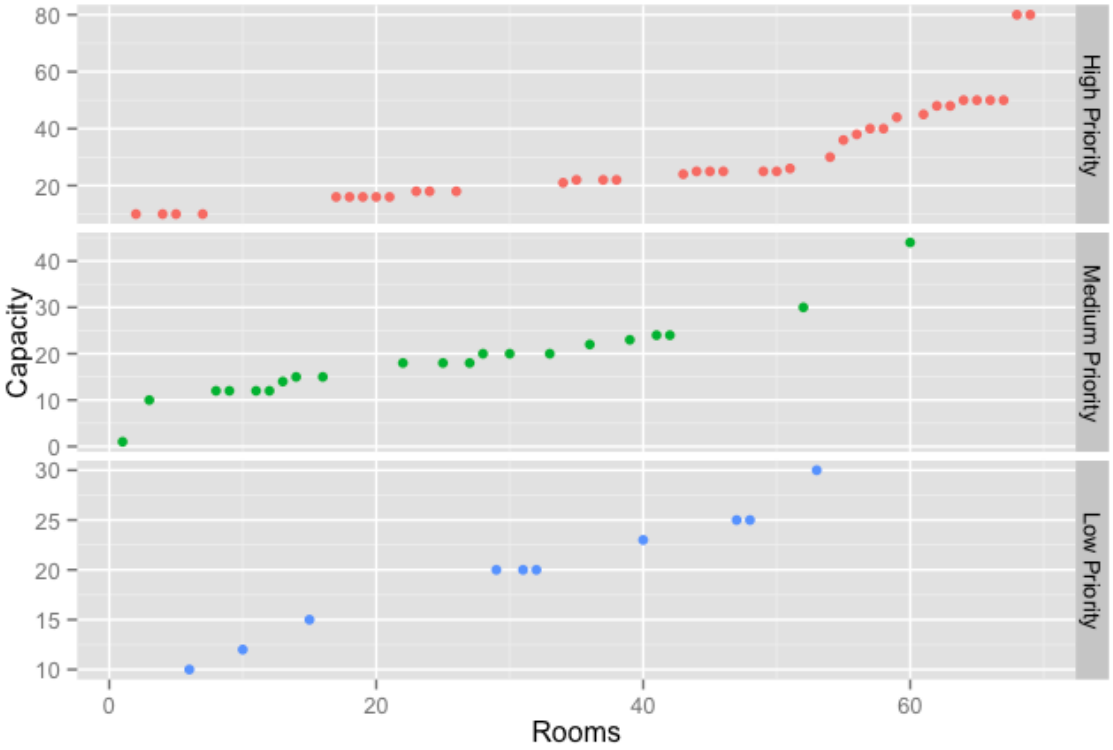

Figure 4.5: Rooms size plotted in an increasing size order by priority

By not using rooms with $\mu = 3$ there will be a reduction of the number in general seats and in computer seats but other seating types, will remain the same. However, these numbers can be tuned slightly upwards if needed . However, if a solution can be achieved using these boundaries they should be used in order to have the best seats available for the students. As mentioned in Section 4.1, the total number of examinations is 25.295. The total number of seats available over the 22 slots with the reduction of rooms with priority $\mu = 3$ is 34078 so there should be enough seats for all of the examinations.

4.2.1. Clusters of buildings

As mentioned in last section there are 14 buildings suitable for examinations. A situation can happen where exams must be split into multiple rooms in the same building or more than one building due to their sizes. When an exam must be split between buildings, the buildings must be chosen carefully to have the examinations at similar locations. Hence, several clusters of buildings are made of buildings which are close to each other. This means, that a split of a course between buildings within a cluster is a better approach than to split a course between two or more clusters. Figure 4.6 shows the clusters of the buildings at the U.I. It consist of three main clusters: *Melurinn*, *Torfan* and *Holtid* and three other clusters. The buildings in the main clusters are located close to each other and some of the buildings are even connected.

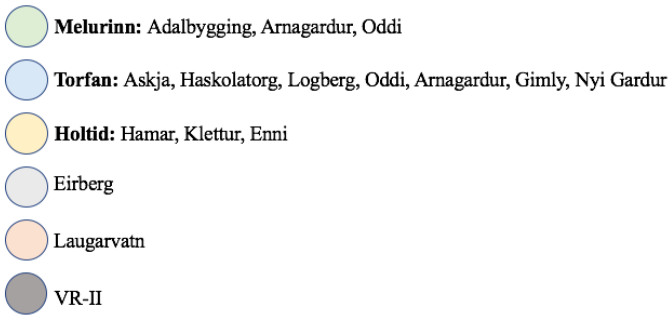

Figure 4.6: Clusters of buildings

Two buildings belong to two clusters, Arnagardur and Oddi. This is done to make the split within clusters easier since those buildings are located at the central university area. Holtid is a cluster of three buildings at Stakkahlid where School of Education is located which is not in walking distance from the central university area. Three other clusters are available but they consists of only one building each. All of those buildings are not located close to any building at the university area and therefore, if an exam is assigned there, it should preferably not be split into other clusters.

4.3. Summary

In this chapter, the data that was collected for the modelling section was analysed in order to get a better understanding of it. Firstly, all of the data regarding the courses and the student's registrations was analysed. In total 25.295 examinations in 524 exams were supposed to be scheduled but behind these examinations were 8901 students. Most of the courses were relatively small with less than 100 students assigned. The connections between the examinations was analysed in Figure 4.3. It shows that most of the courses are connected in one way or another, making the scheduling itself difficult. Similarly, the seats and the buildings data was analysed. From there, it can be seen that there are 4 different seating types available. The seats are spread among 14 buildings and multiple rooms. Not all of the rooms are suitable for examinations and therefore each room receives a coefficient regarding how suitable it is for examinations. In the end of the chapter, clusters of buildings were presented for the problem since it can be avoidable to split an exam between multiple buildings due to their sizes. Each cluster consists of buildings close by each other so if an exam must be split between buildings it is split within a cluster rather than two or more clusters. All of the data has now been investigated and analysed and in the following chapter, the modelling will take place using mixed integer programming.

5. Mixed integer programming models for the University of Iceland

In Chapter 3, the key conditions and objectives that need to be considered when formulating the problem was discussed. In the next sections, the modeling of the constraints and the objectives will take place. The model that will be designed will be a two phase approach similar to Leong and Yeong (1990) where the first phase will be the examination scheduling and the second phase will be the room scheduling. Mixed integer programming (MIP) will be used to formulate the problem. The chapter will be divided into two main Sections 5.1 and 5.2 where the constraints and the objectives will be modelled for each phase.

5.1. Phase I: Examination scheduling

In this section, the first phase of the model will be designed. The section will be divided into multiple subsections. In Subsection 5.1.1, the most general constraints needed for the examination scheduling will be formulated. Subsection 5.1.2 will present the soft constraints of the model while Subsection 5.1.3 will present the constraints that guarantee feasibility for Phase II. Lastly, Subsection 5.1.4 will present the objective function.

5.1.1. General constraints

In this section, all of the hard constraints for the problem will be formulated and a brief description will be given about each one of the constraints.

1. **Fix courses to time slots**
   In general, teachers or departments may require specific time slots T for their courses, therefore adding a constraint that fixes a course to a specific time

slot is necessary. This constraint also gives the opportunity to put more than one time slot for a course if the teachers or the departments request a specific period for a course. For example, a course can require being assigned in the first half of the examination period or the second, specific date, slot, only in the afternoons, in the mornings and so on.

$$\sum_{e \in F_c^f} x_{c,e} = 1 \qquad \forall c \in \mathcal{C} \tag{5.1}$$

2. **Course only assigned to one time slot**
   Each exam should only be assigned to one time slot. Therefore a constraint is added to the model such that each exam is only assigned once.

$$\sum_{e \in \mathcal{E}} x_{c,e} = 1 \qquad \forall c \in \mathcal{C} \tag{5.2}$$

3. **No student clash**
   Students should not take two exams at the same time. A hard constraint is therefore added that does not allow any student to have clashes in any time slot. However, violation of this constraint is allowed if departments have fixed two courses in the same time slot. Occurrences like that are reviewed by the director of examinations and rejected if they are unreasonable.

$$x_{c_1,e} + x_{c_2,e} \leq 1 \tag{5.3}$$

   $\forall c_1 \cup c_2 \in \mathcal{C}, \forall e \in \mathcal{E}$ where $c_1 < c_2$, $|F_{c_1}^f| \neq 1 \wedge |F_{c_2}^f| \neq 1$, $M_{c_1,c_2} \neq 1$ $A_{c_1,c_2} > 0$.

4. **Courses with common questions assigned at the same time**
   Since some courses have questions in common, a constraint must be added so they have the same assignment.

$$x_{c_1,e} = x_{c_2,e} \tag{5.4}$$

   $\forall c_1 \cup c_2 \in \mathcal{C}, \forall e \in \mathcal{E}$ where $c_1 < c_2$, $M_{c_1,c_2} = 1$.

5. **Not two exams the same day**
   To give students as much time as possible to prepare for each examination, occasions where a student is facing two exams during the same day should not

occur at all. However, it is not guaranteed that it is possible to satisfy such conditions for each student since students are often allowed to choose courses between departments and schools. Hence, for tolerance θ, no students should be taking two examinations in the same day above that tolerance. A violation of this constraint is therefore allowed at some rate, thus θ should be chosen as low as possible. Situations can also happen where this constraint can also be violated if teachers or departments have chosen some specific slots for their courses and therefore forced such occurrences.

$$x_{c_1,e} + x_{c_2,e} + x_{c_1,e+1} + x_{c_2,e+1} \leq 1 \tag{5.5}$$

$\forall c_1 \cup c_2 \in \mathcal{C}, \forall e \in \mathcal{E}_d$ where $c_1 < c_2,\ M_{c_1,c_2} \neq 1,\ A_{c_1,c_2} > \theta, |F^f_{c_1}| \neq 1 \wedge |F^f_{c_2}| \neq 1.$

6. **Not two exams in 24 hours i.e. afternoon and morning after**
   Similarly to constraint 5.5 students should not be facing two exams within 24 hours. This means that if a student is having an exam in the afternoon, he should not be having an exam the morning after. However, some degree of violation of this constraint is allowed at the rate of tolerance θ as previously described for constraint (5.5), since it might not be possible to satisfy this condition for all students.

$$x_{c_1,e-1} + x_{c_1,e} + x_{c_2,e-1} + x_{c_2,e} \leq 1 \tag{5.6}$$

$\forall c_1 \cup c_2 \in \mathcal{C}$, $\forall e \in \mathcal{E}_d$ where $c_1 < c_2,\ M_{c_1,c_2} \neq 1,\ A_{c_1,c_2} > \theta$, $\mathcal{H}_e \neq 1, |F^f_{c_1}| \neq 1 \wedge |F^f_{c_2}| \neq 1.$

7. **Have a day off without examinations**
   Students should not be facing two exams two days in a row in order to give students minimum one day off for each exam in order to prepare for an exam. Similarly to constraints (5.5) and (5.6) a violation of this constraint must be allowed at some rate of ϕ which should be chosen as low as possible. Here, the constant ϕ is the lowest number of students in common between courses c_1 and c_2 which are allowed to not have a free day before. Choosing the constants too low can lead to an unfeasible solution.

$$\begin{aligned} x_{c_2,e-2} + x_{c_2,e-1} + x_{c_2,e} + x_{c_2,e+1} + x_{c_1,e-2} + x_{c_1,e-1} \\ + x_{c_1,e} + x_{c_1,e+1} \leq 1 \end{aligned} \tag{5.7}$$

$\forall c_1 \cup c_2 \in \mathcal{C}$, $\forall e \in \mathcal{E}_d$ where $c_1 < c_2$, $M_{c_1,c_2} \neq 1 and\ A_{c_1,c_2} > \phi$,$\mathcal{H}_e \neq 1$, $e > 2$, $|F^f_{c_1}| \neq 1 \wedge |F^f_{c_2}| \neq 1$.

8. **Maximum number of students assigned to each time slot**
   Since U.I has limited seat capacity for each slot a constraint must be added such that the number of students in each slot does not exceed the seating capacity of M^t_e. This constraint is a hard constraint that can not be violated at all.

$$\sum_{\substack{c \in \mathcal{C}: \\ c \notin \mathcal{C}_m}} x_{c,e} S_c \leq M^t_e \qquad \forall e \in \mathcal{E} \tag{5.8}$$

9. **Just one large course in a time slot**
   Due to a limitation on large rooms, a constraint must be added such that only one large course is assigned in each slot. Here, the question arises what a large course is. Hence, a boundary is added to constraint 5.9 with the tolerance α which indicates the number of students which are considered to be a large course. The tolerance α is chosen as the largest number as possible. As α decreases, the model will get infeasible since it will be impossible satisfying this boundary unless the examination period is extended. To have more than one large course in a slot is a hard task to solve in the room scheduling.

$$\sum_{\substack{c \in \mathcal{C}: \\ S_c > \alpha \\ |F^f_c \neq 1|}} x_{c,e} \leq 1 \qquad \forall e \in \mathcal{E} \tag{5.9}$$

5.1.2. Soft objective constraints

In this section, the soft objective constraints for Phase I are modelled. Each one of them holds a variable which will be used in the objective function that will be formulated in Section 5.1.4.

1. **Rest day before**
   This constraint is similar to constraint (5.7) but the constraint is made soft by adding the variable P_{c_1,c_2} which indicates if a course will have a day off before

a scheduled exam.

$$x_{c_2,e-2} + x_{c_2,e-1} + x_{c_2,e} + x_{c_2,e+1} + x_{c_1,e-2} + x_{c_1,e-1} + x_{c_1,e} + x_{c_1,e+1} - 1 \leq P_{c_1,c_2} \tag{5.10}$$

$\forall c_1 \cup c_2 \in \mathcal{C}$, $\forall e \in \mathcal{E}_d$ where $c_1 < c_2$, $M_{c_1,c_2} \neq 1$, $A_{c_1,c_2} > 0$, $\mathcal{H}_e \neq 1$, $e > 2$.

2. **Not two exams the same day**
   Students should not face two examinations in the same day. Therefore, constraint (5.6) is added to the model. Here, the soft constraint includes the variable I_{c_1,c_2} which is a indicator of which courses are having two examinations in the same day.

$$x_{c_1,e} + x_{c_1,e+1} + x_{c_2,e} + x_{c_2,e+1} - 1 \leq I_{c_1,c_2} \tag{5.11}$$

   $\forall c_1 \cup c_2 \in \mathcal{C}$, $\forall e \in \mathcal{E}_d$ where $c_1 < c_2$, $M_{c_1,c_2} \neq 1$, $A_{c_1,c_2} > 0$.

3. **Not two exams in 24 hours**
   Students should not face two examinations in a row even if it is the same day or less than 24 hours as a consequence equation (5.6) was added to the model. Here, a soft constraint is added with the variable U_{c_1,c_2} which indicates the courses having two examinations within 24 hours.

$$x_{c_1,e-1} + x_{c_1,e} + x_{c_2,e-1} + x_{c_2,e} - 1 \leq U_{c_1,c_2} \tag{5.12}$$

   $\forall c_1 \cup c_2 \in \mathcal{C}$ where $c_1 < c_2$, $M_{c_1,c_2} \neq 1$, $A_{c_1,c_2} > 0$, $e \in \mathcal{E}_d$, $\mathcal{H}_e \neq 1$.

4. **Distribute computer exams equally**
   In general, the spread of computer-based examinations should be spread as evenly throughout the examination period. Therefore, the variable J is added as a soft objective upper bound to the constraint.

$$\sum_{c \in \mathcal{C}_c} x_{c,e} S_c \leq J \qquad \forall e \in \mathcal{E} \tag{5.13}$$

5. **Smallest number of students assigned to any time slot**
   In order to make the distribution of the students as equal in each time slot over the examination period a soft objective lower bound constraint is added.

This constraint counts the smallest number of students at any time slot E by the number Z which is the smallest number of students examined for any time slot.

$$\sum_{\substack{c \in \mathcal{C}: \\ c \notin \mathcal{C}_m}} x_{c,e} S_c \geq Z \qquad \forall e \in \mathcal{E} \tag{5.14}$$

5.1.3. Constraints to guarantee feasibility for Phase II

Since the problem is split into a two phase approach, constraints must be added such that feasibility will be achieved for Phase II but in Phase II, the room scheduling takes place. In the room scheduling, the courses are assigned into suitable rooms regarding the rooms they require. The vast majority of the courses include some proportion of students that require special rooms due to disabilities and therefore can not be assigned to general rooms. A more detailed description will be in section 5.2, where the formulation of the model for Phase II will take place. The next constraints that will be added to the model will all be regarding different capacities. Previously, constraint (5.8) has been added to the model as such, but that constraint alone does not guarantee a feasible solution for Phase II.

1. **Maximum number of special students in special rooms**
   Some students may have some disabilities and therefore need to be assigned to special rooms. The type of disabilities can be as many as the students but the vast majority of the special students only require extended examination time and therefore they can be assigned to general special rooms. These special rooms have the total capacity of M_e^s seats that can be assigned to in each slot and therefore constraint (5.15) is added to the model.

$$\sum_{c \in \mathcal{C}_s} x_{c,e} S_c^s \leq M_e^s \qquad \forall e \in \mathcal{E} \tag{5.15}$$

2. **Maximum number of special students in special computer rooms**
   Sometimes, courses require that their examination will be computer based. Therefore, a constraint must be added such that the students with disabilities are assigned to special computer rooms with the upper bound M_c^s of students that can be assigned to each time slot. Constraint (5.16) is therefore added to

the model satisfying these conditions.

$$\sum_{c \in \mathcal{C}_c} x_{c,e} S_c^s \leq M_c^s \qquad \forall e \in \mathcal{E} \tag{5.16}$$

3. **Maximum number of students in computer rooms in each time slot**
   Some courses require their final exams to be computer based. Therefore, a capacity constraint is added such that the total number of general students does not exceed the capacity of available seats in the computer rooms.

$$\sum_{c \in \mathcal{C}_c} x_{c,e} S_c \leq M_c \qquad \forall e \in \mathcal{E} \tag{5.17}$$

4. **Maximum number of students assigned to Stakkahlíð**
   Commonly, the university area can be divided into two main parts, the central university area and Stakkahlíð where the School Of Education is located. In most cases, the courses that belong to the School of Education should be assigned to the Stakkahlíð while other courses should only be assigned to the buildings at the central university area since the distance between those areas are far from each other. Therefore, constraint (5.18) is added to the model.

$$\sum_{c \in \mathcal{C}_e} x_{c,e} S_c \leq M_h \qquad \forall e \in \mathcal{E} \tag{5.18}$$

5.1.4. Objective function

In this section, the objective function will be formulated. In Section 5.1.2, soft objective constraints for the model were formulated but all of the constraints consisted of variables such that the violation of those constraints could happen. In order to keep that violation to a minimum, all of the variables are added to the objective function with weights w_n.

The first part of the objective 5.19 is used to minimize the computer exams in each slot since it is difficult to assign all of the seats at the same time. Part 5.20 of the objective is used to maximize the students assigned to each slot in order make the student assignment into each slot to be as equal as possible.

The last part of the objective function consists of the three key elements. Part 5.21 is used to minimize the total number of students assigned to two exams in the same

day. Similarly, part 5.22 is added such that students do not go to exams within 24 hours and the third part is a minimization of 5.23 which is the total number of students which do not receive one day off before an exam.

$$\min_{\vec{z}}$$

$$w_1 J \tag{5.19}$$

$$- w_2 Z \tag{5.20}$$

$$+ w_3 \sum_{c_1 \in \mathcal{C}} \sum_{c_2 \in \mathcal{C}} I_{c_1,c_2} A_{c_1,c_2} \tag{5.21}$$

$$+ w_4 \sum_{c_1 \in \mathcal{C}} \sum_{c_2 \in \mathcal{C}} U_{c_1,c_2} A_{c_1,c_2} \tag{5.22}$$

$$+ w_5 \sum_{c_1 \in \mathcal{C}} \sum_{c_2 \in \mathcal{C}} P_{c_1,c_2} A_{c_1,c_2} \tag{5.23}$$

The weights should be chosen based on their importance. Hence, the weights should be chosen in the following way: $|w_2| < w_1 < w_4 < w_3$ and $w_1 = w_5$. A more detailed discussion on how the weights are chosen can be found in Section 6.1.

5.2. Phase II: Assignment of courses to rooms

In this section, Phase II of the model will be formulated. The main purpose of Phase II is to assign courses that were previously assigned to time slots in Phase I into suitable rooms. The main goal is to assign as few rooms to each course as possible while spreading the students of each class over multiple rooms in order to uniformly distribute the students between rooms. At the same time at least two courses will be assigned to each room for general students while up to 6 can be assigned for special students due to regulations for the examinations. Similarly, the model will try to maximize that courses will be assigned to their preferred and required buildings.

This section will be divided into two main sections. In Section 5.2.1 all of the constraints needed for the model are formulated and a brief description is given. In Section 5.2.2 the objective function for the model will be formulated and described.

5.2.1. Constraints

In this section the formulation of the constraints takes place. Here, 18 constraints will be formulated and each one of them will be described.

1. **Force a solution**
   Sometimes the director of examinations wants to fix a certain room or rooms for a course. Therefore, constraint (5.24) is assigned to the model so that it is possible.

$$h_{c,r} = h^f_{c,r} \qquad \forall c \in C, \quad r \in \mathcal{R}, \quad h^f_{c,r} > 0 \tag{5.24}$$

2. **Assign all students**
   In order to make sure that all students that are registered to course c are assigned to rooms, constraint (5.25) is added. Due to a limitation of room capacity as can been in Figure 4.4 in Chapter 4, a split of each course into multiple rooms is necessary. This constraint allows this split into multiple rooms while satisfying that each student will be assigned.

$$\sum_{r \in \mathcal{R}} h_{c,r} = S_c \qquad \forall c \in C \tag{5.25}$$

3. **Assign special students to special rooms**
   In general, the students at the university are classified into two groups. One group of students that do not have any disabilities and the other with disabilities (let's call them special students). Special students must be assigned into special rooms since they might need extended time or other requirements for their examination and therefore they can not be assigned into general rooms. Hence, constraint (5.26) is added to the model to ensure all special students are assigned to special seats S_c.

$$\sum_{r \in R_s} h_{c,r} = S_c^s \qquad \forall c \in C_s : c \notin C_c \tag{5.26}$$

4. **Assign students to computer rooms**
   As mentioned earlier, students can be classified into two groups: special students and general students. Within each group, are two subgroups that require computers for their examination. These examinations can only take place in computer rooms which are classified into general and special computer rooms. Therefore, constraints (5.27) and (5.28) must be added to the model respectively to ensure that exams that need computers are only assigned to the computer rooms.

$$\sum_{r \in \mathcal{R}_c^s} h_{c,r} = S_c^s \qquad \forall c \in C_s : c \in C_c \tag{5.27}$$

$$\sum_{r \in \mathcal{R}_c^g} h_{c,r} = S_c - S_c^s \qquad \forall c \in C_c \tag{5.28}$$

5. **The rest of the students should be in other rooms**
   In the last constraints, all of the students that needed to be examined in computer rooms or in special rooms were assigned to appropriate rooms. Therefore, the remaining group of students are assigned to general rooms.

$$\sum_{r \in \mathcal{R}_g} h_{c,r} = S_c - S_c^s \qquad \forall c \notin C_c \tag{5.29}$$

6. **Room capacity**
   At the U.I the exams can take place in multiple rooms at the same time with multiple courses assigned to each room. Therefore, adding a constraint such that the number of students assigned to room r does not violate the room capacity of each room is essential.

$$\sum_{c \in \mathcal{C}} h_{c,r} \leq \mathcal{R}_r \qquad \forall r \in R, \quad \forall e \in E_s \tag{5.30}$$

7. **Courses in rooms**
   For this constraint, the binary variable $w_{c,r}$ is introduced to the model, indicating that if a course c is assigned to a room r it receives the value 1 otherwise 0. In order to connect the binary variable $w_{c,r}$ to the integer variable $h_{c,r}$, the *Big-M* approach is used where M is a big constant hindering violation of the constraint. Here, the right hand sight of equation (5.31) corresponds to M.

$$h_{p,r} \leq w_{c,r} \sum_{c \in \mathcal{C}} S_c \qquad \forall p \in C, \quad \forall r \in \mathcal{R} \tag{5.31}$$

8. **Exams with different duration**
   Most of the exams at U.I require 3 hours in total for their examination. However, some courses may require shorter or longer examination time. Hence, it would be inconvenient for the students to assign courses that have different duration into the same room. Constraint (5.32) eliminates occasions like that to happen by only allowing exams with the same duration to be in the same room.

$$w_{c_1,r} + w_{c_2,r} \leq 1 \qquad \forall c_1, c_2 \in C, \quad c_1 < c_2 \quad \forall r \in \mathcal{R}, \quad d_{c_1} \neq d_{c_2} \tag{5.32}$$

9. **Not too many rooms**
   In order to keep the split of an exam into rooms at minimum, a constraint must be added such that a course is not split into more than one room unless 12 or more students are assigned to the course. If this constraint is not added to the model, high spread of the students between rooms is possible. Hence, occasions could happen such that a course is split into n number of rooms that could be equal to the number of students in the course.

$$\sum_{r \in \mathcal{R}_g} w_{c,r} \leq 1 \qquad \forall c \in C, \quad S_c - S_c^s \leq 12 \tag{5.33}$$

10. **Not too few students**
    Since courses can end up in many rooms they should have at the minimum 12 students assigned to the course for a two way split. This will avoid the possibility of putting courses as singles in rooms.

$$\sum_{r \in \mathcal{R}_g} h_{c,r} \geq w_{c,r} \min\{12, \quad \min\{\frac{S_c - S_c^s}{2}, \quad (S_c - S_c^c) - \sum_{\substack{rr \in \mathcal{R}: \\ h^f_{c,rr} > 0}} h^f_{c,rr}\}\} \tag{5.34}$$

    $\forall c \in \mathcal{C}$ and $r \in \mathcal{R}$ where $S_c - S_c^s \geq 12$.

11. **Not too many courses in a room**
    At the U.I the teacher of each exam visits each room where their exam takes place two times during a three hour exam to answer questions regarding the exam from students. This can lead to interruptions for students as the number of courses increase in a room. However, the larger the room is, less interruption will be for the students. Therefore, constraint (5.35) is added to the model such that most 3 exams are assigned to a room with room capacity over 20 seats otherwise 2.

$$\sum_{c \in C} w_{c,r} \leq \begin{cases} 3 & \mathcal{R}_r \geq 20 \\ 2 & \text{otherwise} \end{cases} \tag{5.35}$$

    $\forall r \in \mathcal{R}$ where $r \notin \{R_s \cup R_c^s\}$, $e \in \mathcal{E}_s$.

12. **Minimum course split for special students**
    Similarly to constraints (5.33) and (5.35) another constraint is added to the model for the special students such that six courses are allowed to be assigned to each room.

$$\sum_{c \in C} w_{c,r} \leq 6 \tag{5.36}$$

    $\forall r \in \mathcal{R}$ where $r \in \{R_s \cup R_c^s\}$, $e \in \mathcal{E}_s$.

13. **Course in building**
Here, the binary variable $wb_{c,b}$ is introduced to the model which receives the value 1 if a course C is in building $\mathcal{B}$ otherwise 0. To link the two variables $w_{c,r}$ and $wb_{c,b}$ together, the *Big-M* method is used where M is a big number.

$$\sum_{r \in \mathcal{V}_r^b} w_{c,r} \leq wb_{c,b} M \qquad \forall c \in C, \quad b \in \mathcal{B} \tag{5.37}$$

14. **Courses with questions in common should be assigned to the same building**
Some courses have questions in common therefore they should be treated as one exam instead of separated exams. Therefore, constraint (5.38) is added to the model so courses that have questions in common are assigned to the same building.

$$wb_{c_1,b} = wb_{c_2,b} \tag{5.38}$$

$\forall c_1 \cup c_2 \in \mathcal{C}$ where $c_1 < c_2$, $b \in \mathcal{B}$, $M_{c_1,c_2} = 1$.

15. **Regular course in building**
For this constraint, the binary variable $wbb_{c,b}$ is introduced. The binary variable receives the value 1 if a course C is in building $\mathcal{B}$ otherwise 0. To link the two variables $w_{c,r}$ and $wbb_{c,b}$ together, the *Big-M* method is used where M is a big number. The main difference between the binary variable $wbb_{c,b}$ and $wb_{c,b}$ is that $wbb_{c,b}$ will only consider the assignments of general students to buildings not others..

$$\sum_{\substack{r \in \mathcal{V}_r^b: \\ r \in \mathcal{R}_g}} w_{c,r} \leq wbb_{c,b} M \qquad \forall c \in C, \quad b \in \mathcal{B} \tag{5.39}$$

16. **Only assign to one building if it is not assigned to building clusters**
If an exam is not assigned to one of the three main clusters of buildings within the University area as presented in section 4.2.1, then it should only be assigned to one building. Here, the variable N_c is introduced to the model which is the total number of buildings that an exam is assigned to.

$$\sum_{\substack{b \in \mathcal{B}: \\ b \notin \mathcal{Q}_g}} wbb_{c,b} + wbb_{c,t} \leq N_c \qquad \forall c \in C, \quad g \in \mathcal{G}, \quad t \in \mathcal{Q}_g \tag{5.40}$$

17. **Room occupation**
    In order to keep track of which rooms are occupied or not, the binary variable wr_r is introduced to the model. If a room is occupied then wr_r is forced to 1, if not it will be 0.

$$w_{c,r} \leq w_r \qquad \forall c \in C, \quad r \in \mathcal{R} \tag{5.41}$$

5.2.2. Objective function

In this section, the objective function will be formulated. The main goal of the model is to assign as few rooms as possible to the courses since opening a new room will cost more for the university since more invigilation staff must be added to shifts. Therefore, each room should have a good room efficiency if it is used.

As mentioned in the problem description in Chapter 3, each course does have a preferred building and also a list of required buildings where they should be assigned if they cannot be assigned in their preferred building. Therefore parts 5.42 and 5.43 are added to the objective function. The preferred building is a list for each course of which building they prefer the most. The model will therefore try to minimize that courses will be assigned elsewhere by part 5.42 in the objective. However, when a course cannot be assigned to it's most preferred building, it will be assigned to a required building in part 5.43. This list is similar to the list described for part 5.42 but includes a set of other options for buildings. Therefore, this will be a step by step penalization if a course will end up in other buildings than on those lists, receiving costs from both parts. Part (5.44) is the general assignment cost of a course to a building. That part along with part (5.45) are added to the objective function in order to assign as few buildings as possible for each course. The main difference between the parts is that part (5.44) minimizes the total number of buildings assigned to a course in general. Part (5.45) of the objective only takes consideration to general courses and minimizes when a course is assigned to buildings which are not in the same clusters of buildings.

To open new room costs extra invigilation staff and therefore it is necessary to have as many rooms free in each time slots. Hence, part (5.46) tries to force the objective to have as many rooms free in each time slot and therefore maximize the room usages of each room that is being used. Since, the director of examinations would like to spread students equally between rooms, part (5.47) is added to the objective in order to maximize the spread of the students into as many rooms as possible. This step is done to balance the total number of students in each room. Typically, this is performed by a quadratic formulation but since a linear programming is used to formulate this problem, it must be done in that way. The last part of the

objective function is part (5.48) which tries to minimize courses to be assigned to rooms with a low priority. All of the rooms are ranked in order how suitable they are for examinations as mentioned in Chapter 4. Therefore it is necessary, in order to use the best rooms for the examinations.

$$
\begin{aligned}
&\min_{\vec{x}} \\
&\quad K_1 \sum_{c\in\mathcal{C}} \sum_{b\in\mathcal{B}} wb_{c,b} \quad b \notin \mathcal{P}_b^c && (5.42)\\
&+ K_2 \sum_{c\in\mathcal{C}} \sum_{b\in\mathcal{B}} wb_{c,b} \quad b \notin \mathcal{R}_b^c && (5.43)\\
&+ K_3 \sum_{c\in\mathcal{C}} \sum_{b\in\mathcal{B}} wb_{c,b} && (5.44)\\
&+ K_4 \sum_{c\in\mathcal{C}} N_c && (5.45)\\
&+ K_5 \sum_{r\in\mathcal{R}} wr_r && (5.46)\\
&- K_6 \sum_{c\in\mathcal{C}} \sum_{r\in\mathcal{R}_g} w_{c,r} && (5.47)\\
&+ K_7 \sum_{c\in\mathcal{C}} \sum_{r\in\mathcal{R}} w_{c,r}\mu_r && (5.48)
\end{aligned}
$$

Here the cost of the objective functions K_n should be chosen in the following order based on importance for the objective: $K_1 = K_2$ and $K_7 < |K_6| < K_3 < K_1 < K_5 < K_4$. A more detailed discussion on how the weight are chosen can be found in Section 6.1.

5.3. Summary

In this chapter, the modelling of the the examination scheduling and the room scheduling took place for the University of Iceland. The key objective of the examination scheduling was to minimize the same day examinations, examinations within 24 hours (afternoon and morning after) and minimize the number of students who could not get one day off before an exam. Two other goals were also set, to have the total number of students assigned to each slot similar and to have the computer based exams as spread as possible. The key objective for the room scheduling was to assign as many exams to their most preferred buildings, if that was not possible they should be assigned to required buildings. Similarly, a course should be assigned

to as few rooms and buildings as possible. Two other objectives were put with a low penalty cost in order to maximize the spread of the students and to use as many rooms which are the most suitable for examinations. Since all of the modelling has taken place for Phase I and Phase II, computational experiments can now be performed and will be done in the following chapter.

6. Computational Experiments

In this chapter, computational experiments of the models that were designed in the last chapter will be performed. The data that will be used was described in Chapter 3. The models were written in MathProg, programming language using Gurobi solver (*version 6.0.2*). All of the runs were performed on a 12 core Intel(R) Core(TM) i7-4930K CPU @ 3.40GHz desktop machine running on Debian GNU/Linux operating system with 32GB internal memory using 6 threads.

The chapter will be divided into three main sections. Section 6.1 will describe the parameters used and how they were chosen. In Section 6.2 the experimental studies will be described for Phases I and II. Lastly, Section 6.3, the computational experiments will be performed and the results will be introduced.

6.1. Parameter settings

In this section, the parameter settings that will be used for the computational experiments in Section 6.2 will be introduced.

1. **The Examination Period**
   The examination period that the computational experiment will be performed on contains $n = 11$ days. Each day contains two time slots where exams can be assigned. Therefore, 22 time slots are available for the examination period.

	Mon	Tue	Wed	Thu	Fri	Sat	Sun
					2-Dec	3-Dec	4-Dec
Week 1					1 2		
	5-Dec	6-Dec	7-Dec	8-Dec	9-Dec	10-Dec	11-Dec
Week 2	3 4	5 6	7 8	9 10	11 12		
	12-Dec	13-Dec	14-Dec	15-Dec	16-Dec		
Week 3	13 14	15 16	17 18	19 20	21 22		

Figure 6.1: The examination period calendar

Since no exams are assigned during the weekends or on holidays, a binary indicator must be added such that if there is a holiday or a weekend before a time slot it will receive the value 1. If there is no weekend or holiday it will be indicated with 0. Therefore, the binary indicator H_e can be formulated as equation (6.1)

$$H_e = \begin{cases} 1, & \text{if } e \in \{1, 2, 3, 4, 13, 14\}. \\ 0, & \text{otherwise.} \end{cases} \tag{6.1}$$

for all $e \in \mathcal{E}$.

2. **Maximum number of seats**
   The total number of seats available at U.I is 1749. That number consist of multiple types of rooms as discussed in Section 4.2. In this section, the upper bounds of each seating type will be presented.

   i) **General seats**
   The upper bound on the total number of seats used for general examinations for each slot M_e^t was chosen to be 1168. The main reason why that number is not higher is as the the number increases it can lead to difficulties when assigning rooms to courses due to various requirements. Although M_e^t is 1343 not all of the seats available in that number, are considered to be suitable for examinations due to their small sizes or other discomforts. Each room receives a coefficient μ which can either be $\{1, 2, 3\}$. When $\mu = 1$, then that room is considered to be good for examinations while $\mu = 3$ would be a bad choice. Similarly, should no course be assigned to Laugarvatn unless it is taught there. Therefore, $M_e^t = 1168$ is considered to be a good upper bound for general seats.

   ii) **Special seats**
   The total number of special seats M_e^s available is 198 and they should all be used for examinations if needed.

   iii) **Computer Seats**
   The total number of computer seats available M_c are 170. Although not all of the seats are good for examinations, constraint 5.13 should spread the load equally over the examination period such that the total number of the total number of seats available is not reached.

   iv) **Special Computer Seats**
   The total number of special computer seats available M_c^s is 38.

v) **Stakkahlíð**
In general, no examinations should be assigned to Stakkahlíð unless they belong to the School of Education since it is not at the central university area. Stakkahlíð consists of three buildings, *Hamar*, *Klettur* and *Enni*. However *Klettur* and *Enni* are considered to be bad choices for examinations and therefore they should not be counted for the total capacity M_h. Therefore the total capacity M_h is 194.

As a result, Table 6.1 is made with all of the capacities available at the university area for all time slots.

Seat type	Name	Number of seats
General seats	M_e^t	1168
Special seats	M_e^s	198
Computer seats	M_c	177
Special computer seats	M_c^s	38
School of education	M_h	198

Table 6.1: Number of seats for each seat type

3. **Tolerance for the number of common students having no free day before an exam**
Occasions where students are not receiving a day off before an exam is allowed to happen by a rate of ϕ for constraint (5.7) where $A_{c_1,c_2} > \phi$. This parameter is chosen to be as small as possible. The smallest number achieved for ϕ was 17 but otherwise it leads to an infeasible solution. Adding this parameter should increase the computational speed since the search space decreases because the model will allow some degree of ϕ to have exams the same day while the objective function will minimize such occasions from happening.

4. **Tolerance for the number of common students having two exams in the same day or in a row**
Similarly to the last paragraph, another tolerance α is added such that it allows students at some rate to be having exams the same day or in 24 for constraints (5.5) and (5.6) where $A_{c_1,c_2} > \alpha$ respectively. Here, the parameter should be chosen as low as possible and such that $\alpha < \phi$. The smallest number achieved was 6. This trick was done to increase the computational speed by allowing some number of students to have two exams the same day or in 24 hours. This decreases the search space and should therefore increase the computational speed.

5. **Large course tolerance**
As discussed in paragraph 9 in Section 5.1.1, just one large course should be assigned to each time slot. One may ask what a large course is. Constraint

(5.9) was added to the model in this context with the conditions that $S_c > \alpha$. Here, the parameter α is unknown but it should be chosen as low as possible. If the parameter α is too low the problem will be infeasible. By trial and error approach the number 260 was the lowest number that could be chosen. Therefore, only one course that contains more than 260 students can be assigned to each time slot.

6. **Costs parameters for the objective functions**

   a) **Phase I**
   The costs parameters $w_1, w_2 \dots w_7$ for the objective function are chosen based on their importance to the problem. In section 5.1.4, the objective function was formulated for Phase I and how the parameters should be chosen based on their importance. The cost parameters are associated with the number of students the model should "sacrifice" in order to improve the solution.

   The lowest cost of the key objectives is put on part 5.23 where the cost w_5 is associated. The cost for w_5 is chosen to be 1. Consequently, the model will try to give all students at least one day off before an exam. Whenever it is not possible to satisfy that condition part 5.22 steps in with the cost of w_4 of 50.

$$w_4 \sum_k x_4^{(k)} = w_5 \sum_k x_5^{(k)}$$
$$\sum_k x_5^{(k)} = \frac{w_4}{w_5} \sum_k x_4^{(k)}$$

   As can be seen, the cost of assigning a student to two examinations within 24 hours will be equal to 50 students having a day off before an exam. Therefore, it can be seen that the objective function will try to put as many students to a one day off before each exam since it will sacrifice 50 students in a rest for 1 student going to exams within 24 hours. The weight w_3 is chosen to carry the cost 100. Therefore, it will sacrifice

$$\sum_k x_5^{(k)} = \frac{w_3}{w_5} \sum_k x_3^{(k)}$$

   100 students with a day off for one student that will have same day examinations. By comparing costs w_4 and w_5 associated with parts 5.22 and 5.21 respectively it can be seen that

$$\sum_k x_4^{(k)} = \frac{w_3}{w_4} \sum_k x_3^{(k)}$$

the objective function will preferably sacrifice 2 students to go to examination within 24 hours instead of putting 1 student in same day examinations.

Two other parts ((5.19) and (5.20)) with costs w_1 and w_2 respectively are also in the objective but are not considered to be as important as the others. As mentioned in Chapter 5.1.4, then $w_1 = w_5$. However, this part of the objective is only correlated to computer examinations in order to minimize the total number of students in each time slot so the examinations can be distributed as equally as possible.

$$J = \frac{w_5}{w_1} \sum_k x_5^{(k)}$$

So this part is as important as part (5.23) and follows the same logic as part w_5.

Part 5.20 tries to maximize the students assigned to each time slot. This part is necessary to have the number students assigned to each slot as equal as possible. However, the weight w_2 receives the lowest weight of the objective or $w_2 = 0,0001$. Since, the cost which is correlated to the variable Z is maximized it is likely that the Z will be equal to the largest number of examinations achieved by satisfying all of the hard constraints for each time slot in the beginning of the optimization, but will get lower over time. Hence,

$$-Z = \frac{w_5}{w_2} \sum_k x_5^{(k)}$$

by changing Z by one student (*let's say that number is* 1160) will be equivalent to sacrifice 9 students with more than one day off before an exam and only give them one day off.

Hence, the objective function will try to give as many students as possible a day off before an exam. When that condition cannot be satisfied it will put the minimum number of students to exams within 24 hours and whenever it is not possible to satisfy that the last resort is to put students

to same day examinations. In the end, it will try to have the number of computer examinations as equally spread over the examination period and similarly it will try to maximize the students assigned to each time slot.

b) **Phase II**

In this section, the cost parameters $K_1, K_2...K_7$ will be chosen for the objective function found in 5.2.2, based on their importance. In general, the cost of assigning a course to a building is associated with the cost parameter K_3 which is correlated with part 5.44 of the objective function. The cost is chosen to be 10 which is one of the lowest costs.

The objective function starts with parts 5.42 and 5.43 which are associated with costs K_1 and K_2 and the costs is chosen to be 20 for both terms. The first part of the objective tries to assign as many courses into their most preferred building since it will only receive the general assignment cost of K_3. As soon as a course cannot be assigned to its preferred building, it will additionally receive the cost K_1 and will be assigned to a required building. Similarly, if a course cannot be assigned to its required buildings, it will receive costs K_1, K_2 and K_3 or in the total of 50. Therefore, the objective will try to assign as many courses to their preferred buildings while using as few buildings as possible.

The largest cost coefficient K_4 is associated with part 5.45. of the objective function. This part does not include buildings assigned for special students since they might need to assigned to another sets of buildings. Therefore, the cost parameter is chosen to be 1000.

$$K_3 \sum_k x_3^{(k)} = K_4 \sum_k x_4^{(k)}$$
$$\sum_k x_3^{(k)} = \frac{K_4}{K_3} \sum_k x_4^{(k)}$$

As can be seen,are 100 general building assignments to preferred buildings are as cost equivalent as the assignment of a course to only one building for general students not in preferred buildings. As previously discussed in Chapter 5.2.2, it is necessary to use as few rooms as possible since opening a new room means additional invigilation staff must be added to shifts. Therefore, the cost K_5 which is associated with part 5.46 of the objective is chosen to be 100.

$$K_3 \sum_k x_3^{(k)} = K_5 \sum_k x_5^{(k)}$$
$$\sum_k x_3^{(k)} = \frac{K_5}{K_3} \sum_k x_3^{(k)}$$

As can be seen, it will be as cost equivalent to have a room free as ten general building assignments to preferred buildings. Therefore, the objective function will try to assign as many courses to rooms in their preferred buildings rather than keeping a room free. The last two parts of the objective are parts 5.47 and 5.48 which are associated with the cost parameters K_6 and K_7. The cost parameter K_6 is chosen to be:

$$\frac{1}{|C_c|} \simeq 2,00 \cdot 10^{-3}$$

This part of the objective tries to spread out the students to multiple rooms in order to equalize the students assigned to each room while also assigning students to rooms which are not suitable for examinations. This part of the objective function is a maximization and therefore, the cost will be negative.

$$K_3 \sum_k x_3^{(k)} = (-K_6) \sum_k x_6^{(k)}$$
$$\sum_k x_3^{(k)} = -(\frac{K_6}{K_3}) \sum_k x_6^{(k)}$$

One room assignment will be equivalent to $-2 \cdot 10^{-4}$ building assignments of a course to a preferred building. Since this part of the objective function has a negative cost then it will decrease the overall cost when used. However, this cost is low and should have negligible effects on the objective function. This part, is more thought as fine tuning. The last cost parameter K_7 is associated with part 5.48 of the objective function which represents the maximization of rooms which are the most suitable for examinations. The cost parameter is chosen to be:

$$\frac{0,1}{|C_c|} \simeq 2,00 \cdot 10^{-4}$$

This part of the objective will try to assign exams to rooms which are the most suitable for examinations. As described in section 4.2, each room receives a coefficient of μ on the interval $1, 2, 3$ which represents their

suitability. This part will therefore try to assign as many examinations to rooms which are suitable for examinations.

$$K_6 \sum_k x_6^{(k)} = K_7 \sum_k x_7^{(k)}$$
$$\sum_k x_3^{(k)} = \frac{K_7}{K_3} \sum_k x_7^{(k)}$$

Therefore, to assign a course to a room which is suitable for examinations is as cost equivalent as to assign a $2 \cdot 10^-5$ preferred building to a course. Clearly, the objective will try to assign as many courses to suitable rooms due to a low cost while trying to use as few buildings as possible.

Therefore, the objective function will try to assign as many courses to rooms with a good room priority. After it will try to assign the courses into few buildings but at the same time assign them to their preferred or required buildings. Sometimes the special students must be assigned elsewhere at the university area the objective function tries to keep the general students in the same building while allowing special students to be assigned elsewhere. By fine tuning the problem, a maximization part is added to spread the students to as many rooms as possible.

6.2. Experimental study

6.2.1. Phase I: Different objectives

The main experiments that will be performed on the model proposed for Phase I, is to investigate different combinations on the objective function for the model. The main investigation is to find the most suitable objective function for the model which gives a good quality solution for students. In order to make a good quality solution, the following goals must be taken into consideration.

(i) Students should not have any clashes in the same time slot

(ii) Students should not go to two exams the same day

(iii) Students should not go to two exams within 24 hours i.e. the afternoon and the morning after

(iv) Students should get at least one day off before each exam

(v) The variation of number of students assigned to each slot must be as equal as possible

(vi) The variation of number of students assigned to a computer based examination should be as equal as possible

In Table 6.2, the different objectives that will be investigated are listed. The + indicates that the following part of the objective function is used while − indicates that part of the objective function was not used. The termination time will be set to 3 hours in all cases with weights for the objective function as $w_1 = w_5 = 1$, $w_2 = 0,0001$, $w_3 = 100$ and $w_4 = 50$. The following weights are based on the importance of each objective for the problem as discussed in last section.

Objectives	J	z	$\sum\sum I_{c_1,c_2} A_{c_1,c_2}$	$\sum\sum U_{c_1,c_2} A_{c_1,c_2}$	$\sum\sum P_{c_1,c_2} A_{c_1,c_2}$
0)	-	-	-	-	-
1)	-	-	+	-	-
2)	-	-	+	+	-
3)	-	-	+	+	+
4)	-	+	+	+	+
5)	+	+	+	+	+

Table 6.2: Different Objectives

Similarly, in order to make a good quality solution for Phase II it is necessary to have as few large courses assigned to each time slot. Having too many large courses assigned in a time slot can make difficulties in the room scheduling. Therefore, this part must not be overlooked.

6.2.2. Phase II: Multiple experiments

Phase II of the model, will use the best solution achieved in Phase I by using the most suitable objective function as can be seen in Table 6.2. The main experiments that will be performed for this phase will be to investigate how well the model is performing on the following segments:

(i) How many rooms are used on average for each course by size?

(ii) How good is the room efficiency when a room is used?

6.3. Computational Results

6.3.1. Phase I

In Table 6.3 the computational results for the different objectives are listed. The runs were terminated after 3 hours if the duality gap was not equal to 0,00%. Afterwards, the three of the most promising objectives will be chosen based on the quality of the solutions and will be run for 60 hours each.

Objective	J	z	$\sum\sum I_{c_1,c_2} A_{c_1,c_2}$	$\sum\sum U_{c_1,c_2} A_{c_1,c_2}$	$\sum\sum P_{c_1,c_2} A_{c_1,c_2}$	Incumbent	Duality Gap	Time [sec]
0)	-	-	608	264	2126	0,00	0,00%	6,82
1)	-	-	**75**	465	2149	7500,00	0,00%	6990
2)	-	-	80	**46**	1981	10100,00	33,17%	10800
3)	-	-	362	128	**1383**	43983,00	87,05 %	10800
4)	-	960	329	131	1439	40888,90	86,17 %	10800
5)	**170**	**933**	206	211	1465	32769,91	82,33 %	10800
3)*	-	-	102	56	1287	14287,00	53,50 %	216000
4)*	-	948	**90**	**43**	1407	12556,90	47,02 %	216000
5)*	**128**	**904**	109	52	**1238**	14865,91	52,30%	216000

Table 6.3: Computational experiments for the model with different objectives[1]

The main purpose is to find the most suitable objective function of the ones investigated. Objectives 0) and 1) are not considered to be suitable due to a high

[1]Models indicated by * are models with 60 hour runs.

number of students not receiving one day off before an exam and due to a high number of courses having exams in the same day. In Figure 6.2 the overall variation between the easiest and the busiest time slot in the examination period can be seen for each objective function. The lower bound for each box indicates the lowest number of students assigned to any time slot in the examination period for each objective whereas the upper bound for each box indicates the maximum number of students assigned to any time slot. If objectives 0) and 1) are compared, it can be seen that there is a high variation between the easiest and the busiest time slot in the examination period and therefore they are not considered suitable objective functions. Objectives 2) and 3) are dissimilar. Objective 2) outperforms objective 3) in the same day examinations and students having exams within 24 hours but the students receiving one day off is better for objective 3). Similarly, from Figure 6.2 it can be seen that objective 3) is better since the gap between the easiest and the busiest day is smaller. By comparing objective 3) and 4) it can be seen that the results are quite similar but the students not receiving one day off is less for objective 3) but at the same time objective 4) has a smaller gap between the easiest and busiest day. At this stage, it is not really clear which objective is the best one. Due to a high number of students not receiving one day off before an exam, objective 2) is not considered to be a good objective. Hence, objective 3), 4) and 5) will be investigated with 60 hour runs.

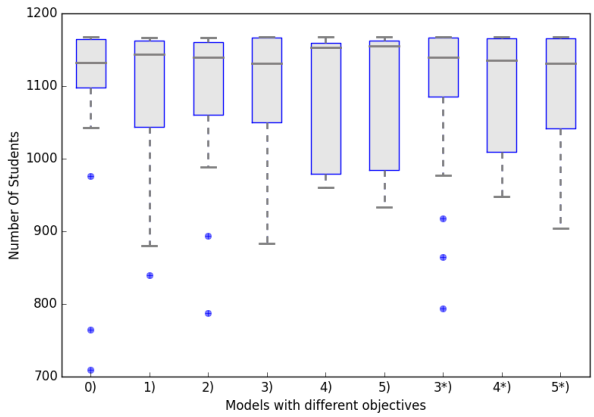

Figure 6.2: Variation between the easiest and the busiest time slot by objective

After the longer runs on the models with the three most promising objectives, it is still not totally clear which of the objectives performs the best. By comparing 3) and 5), it can be seen that they perform similarly. The main difference lies between the students not receiving one day off before their examination which is 49 students less for objective 5). However, the gap between the easiest and the busiest day is the highest for objective 3) compared to the other objectives. This makes objective 3) not a good choice for an objective since the variation between the easiest and the busiest day must be as equal as possible. By comparing objective 4) and 5) it can be seen that objective 4) outperforms objective 5) for the same day examinations and

examinations within 24 hours. However, students not receiving one day off before an exam is worse for objective 4). This makes objective 4) not a good choice for the problem since it is better to give more students having one day off before an exam. One of the main goals for the objective is to have the computer based examinations as equal as possible over the examination period. By looking at Figure 6.3, it can be seen that objective 5), performs the best. The other two objectives have big spikes over the examination period and do not spread the computer based examinations equally as objective 5). Consequently, the most convenient objective function for the problem is chosen to be objective 5).

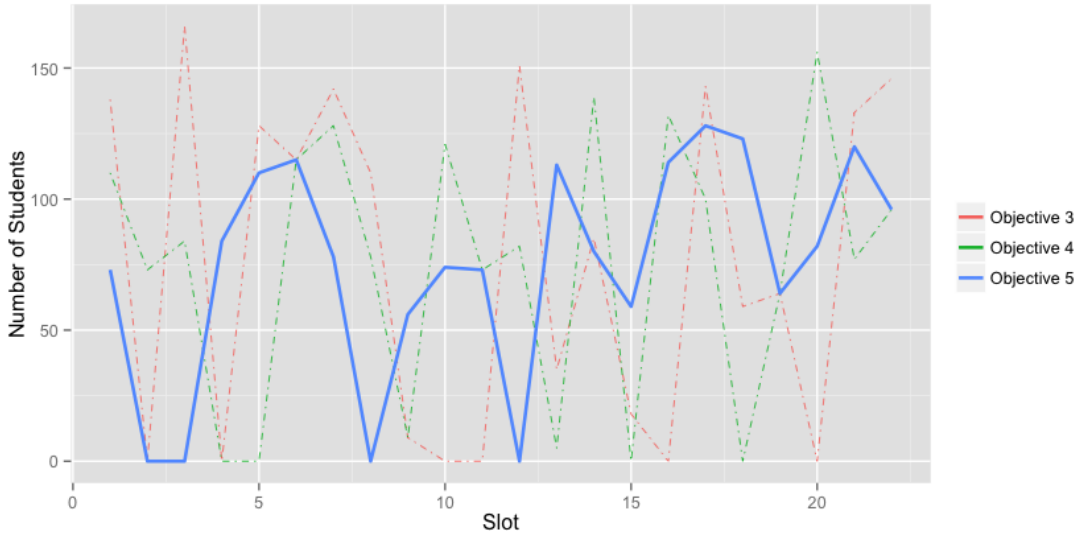

Figure 6.3: The spread of the computer based examinations

The results in Table 6.3 are based on the real data for the autumn semester of 2016. Therefore, each individual course may require a specific time slot or period during the examination period. Hence, if a group of students is assigned to two courses that both require the same time slot, the same day or the next day after, this group of students will be forced to go to these examinations within a short time if overseen by the director of examinations or other. Hence, by allowing each individual course to require a specific time slot or period can damage the solution but at the same time, such conditions decrease the search space and therefore the problem will be less computationally expensive. Some conditions are hard such as courses requiring only one day or perhaps one time slot while others are more flexible requiring the first week or the second one. The tighter these conditions are for the courses, the solution will be damaged. By looking at Figure 6.4, the sparse matrix can be seen for the solution achieved by objective 5. If an optimal solution would be achieved then an empty white stripe (*the white area on the figure*) could be drawn by the diagonals. Although that situation is satisfied mostly, there are still conflicts in the table since there are dots inside this white stripe. The total number of non-zeros elements in this conflict matrix is 6654 and it's density is calculated to be $2,42\%$.

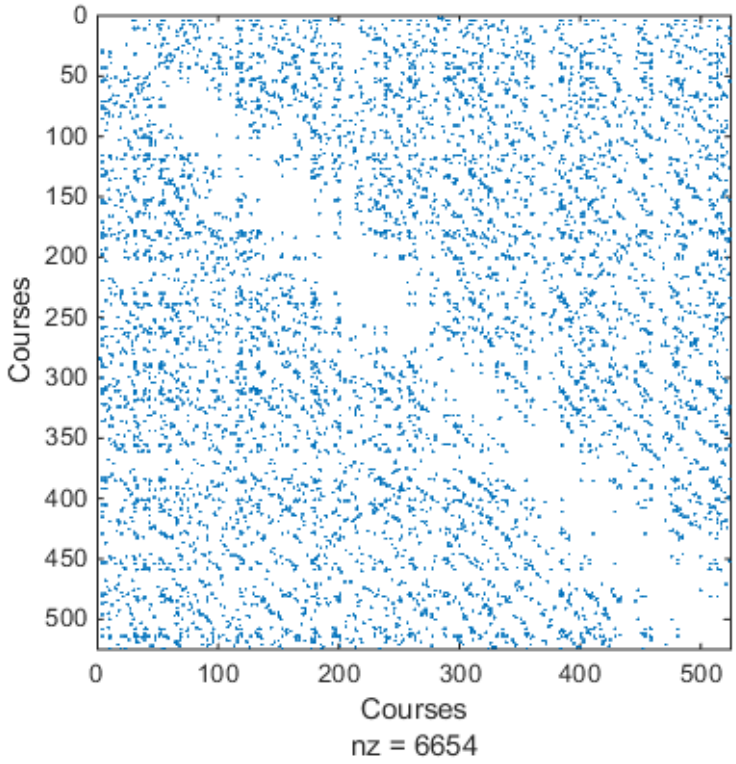

Figure 6.4: Sparse matrix for the solution of objective 5)

6.3.2. Phase II

As mentioned in Section 6.3.1, the best objective function was chosen to be objective 5) for the model. Hence, a solution after a 60 hour run using that objective function for the model will be used as an input for Phase II. Since each time slot is an independent assignment problem, the problem will be split into 22 sub-problems that are solved individually. This is done in order to get a solution quickly for each time slot but this method is similar to the director of examinations uses. The time limit for each run will be set to 600 seconds. That time is considered a reasonable time to solve the room scheduling. Similarly, 6 threads will be used for all of the runs.

The first investigation performed, was to check if there are any trends in the assignment of the total number of rooms assigned to each course. In Figure 6.5 the courses are plotted by their number and the number of rooms assigned to the course. The bubbles on the figure indicate the total number of students enrolled in the course. As can be seen in the figure, when the courses are small in size it won't be assigned to many rooms. However, as the size of the course increases, the assignments will be more interval based, e.g. a course with 100 students can be assigned to 3 or 6

rooms. The maximum number of rooms assigned to a course equals to 13. What makes this high variation of the courses splitting into multiple rooms happens when there are two or more large courses assigned to the same time slot and the fact that there are limited number of rooms with large capacities.

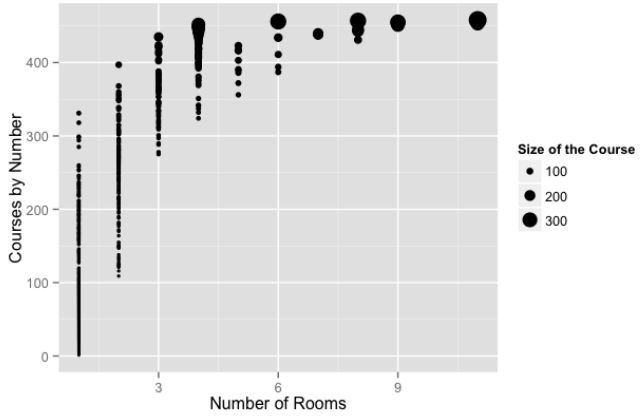

Figure 6.5: Number of rooms by the size of the courses

As formulated for Phase I, only one large course should be assigned to each time slot. The largest number that was achieved was $\alpha = 260$ as described in Section 5. However, as can be seen in Figure 6.6 there are on occasion two or more large courses assigned to the time slot. Therefore, there is a limitation to constraint 5.9 since multiple courses with students above α can be assigned to each time slot. By looking at Figure 6.6 it can be seen that time slot 2 can be difficult for the room assignments since 3 courses with more than 250 students are assigned to that time slot. Hence, some of the large courses will end up in multiple rooms between multiple buildings.

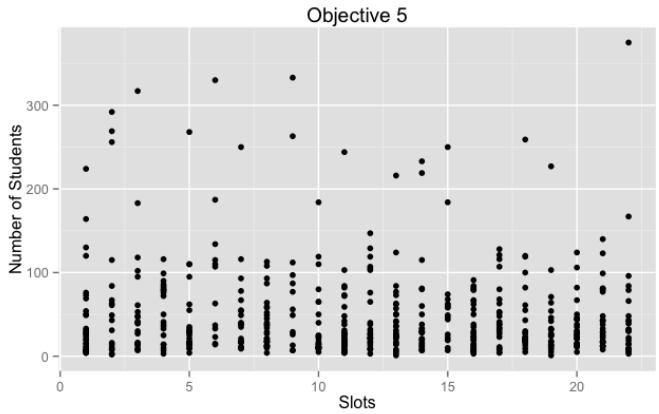

Figure 6.6: Sizes of the courses assigned to each slot

In figure 6.7 the room efficiency for each room used is calculated and plotted. Each dot indicates how many times each room was used and similarly, the total efficiency

can be seen. The total room efficiency is calculated as the total number of seats used in each room in a given slot divided by the total number of seats in the room. From the figure, can be seen that most of the rooms have a really good room efficiency (re) but most of the rooms have **re** between $80\% - 100\%$, some between $60\% - 80\%$ but a minority of the rooms have less than 60%. To fill the room as much as possible is necessary since fewer rooms will be used and therefore less invigilation staff is required.

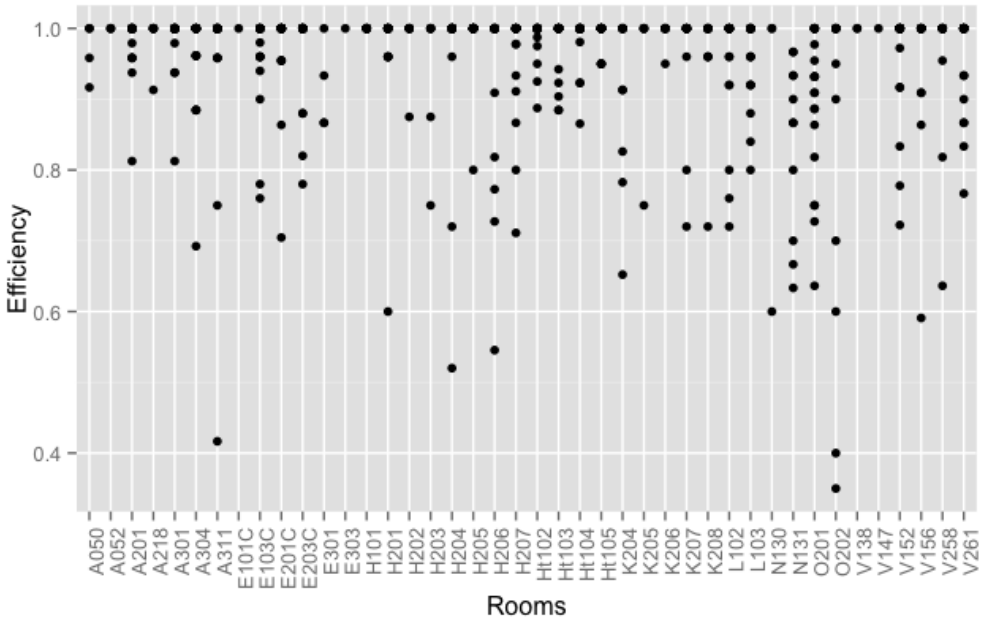

Figure 6.7: The room efficiency of each room used

In Figure 6.8, the total number of students assigned to each time slot is plotted. In the plot, three time series can also be found where each one of the series represents the total number of seats used in each slot for each room priority as described in Section 4.2. From the figure, it can be seen that rooms with priority coefficient $\mu = 1$ are mostly used but the rooms with priority coefficient $\mu = 2$ and $\mu = 3$ are used less and are used almost equally. As mentioned earlier in Section 4.2, rooms with priority coefficient $\mu = 3$ are not well suitable for examinations. However, in the cases when these lower priority rooms are being used, the main reason is that it is better to use lower priority seats than to split a course into more rooms or between buildings as discussed in Section 6.1.

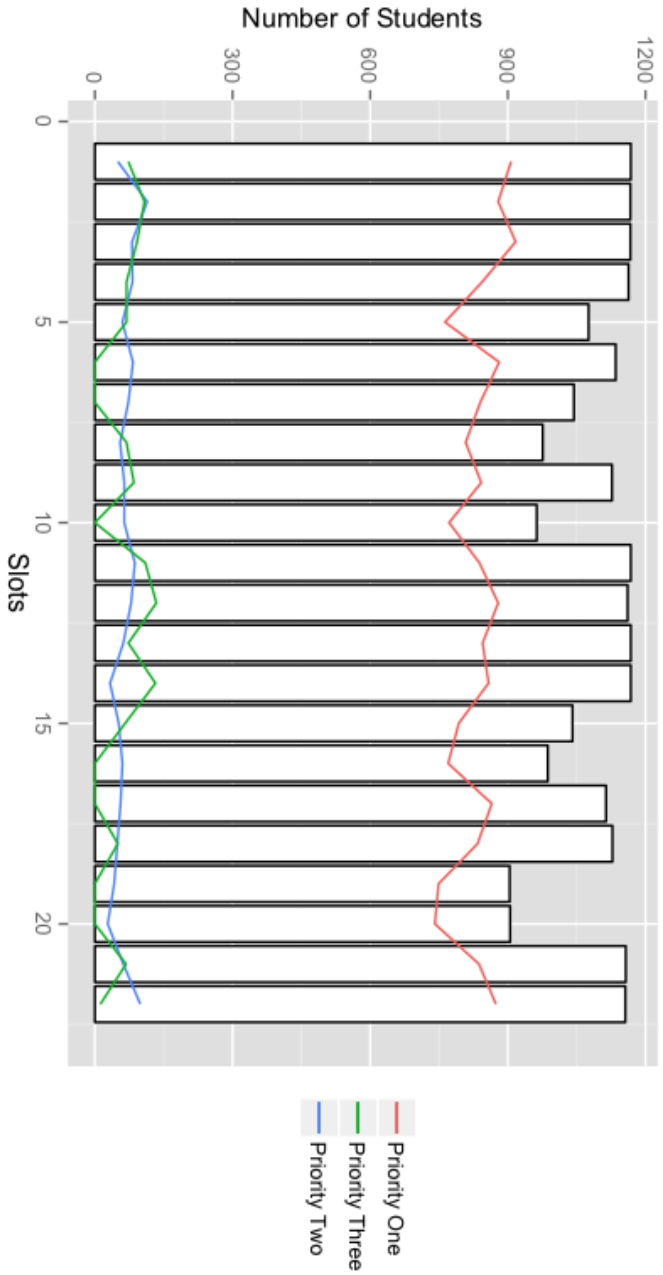

Figure 6.8: The number of each priority seat category versus number of students

6.4. Summary

In this chapter the computational experiments took place. Firstly, all of the parameters were tuned and all of the seating capacity on different seating type was presented. Similarly, where the cost parameters for the objectives chosen for Phase I and Phase II regarding their importance. After the experimental study took place for Phase I and Phase II. The main experiments performed for Phase I was to find the best objective function for the model. Five different type of models were investigated and was objective 5) chosen to be the best one since it gave the best overall result. The main experiments for Phase II were to see how many rooms are used on average for each course and how good the room efficiency was. The model performed well and had a good room efficiency in general. Similarly, it could be seen that most of the courses require $1 - 3$ rooms but they usually have less than 100 students. However, no trends can be seen on how many rooms are assigned for larger courses. For example, a course with 300 students can be assigned to 4-13 rooms. This variance may be explained by assigning two or more large courses in the same time slot. Since the computational experiments showed that Phase I was computationally expensive, a heuristic approach will be proposed in the following chapter.

7. Heuristic for Phase I

In Chapter 6, the computational experiments were performed on different combinations on the objectives for Phase I. The best objective that was chosen for the model was objective 5). As can be seen in Table 6.3, the problems are computationally expensive and therefore a question arises if there is a way to solve the problem in a more reasonable time.

The examination scheduling is a hard task to solve. Not all of the exams equals in difficulty since they differ in sizes and conflicts between other courses. At the U.I, students are often permitted to select elective courses, which makes the examination scheduling a lot harder. As a consequence, a pre-processing process of making a list based on the difficulty of each course ahead to their assignment to time slots must take place but such technique has been used widely in the literature. The method that will be used is based on the work in Carter et al. (1996) paper or the largest weighted degree. Largest weighted degree (LWD) is a low-level constructive heuristics which is based on the largest degree of conflicts each course has and largest enrollment to courses (Carter et al., 1996; Broder, 1964). This method is similar to the one that was used previously when constructing the examination timetable manually and therefore chosen as a suitable approach for the problem and is also easy to implement.

Let A be a symmetric coincidence matrix between courses c_1 and c_2 where $c_1 \cup c_2 \in \mathcal{C}$. Each edge (i, j) in the matrix, represents the the total number of common number of students which are assigned to both courses i and j. When $i = j$ the edges represent the total number of students assigned to the course. Therefore, the LWD can be calculated as in Equation (7.1).

$$LWD_i = \sum_{c_2 \in \mathcal{C}} A_{c_1,c_2} \qquad \forall c_1 \in \mathcal{C} \tag{7.1}$$

Here, the LWD is an indicator of how difficult an assignment is for every course c. By using this method the enrollment of students to a course is associated with the number of conflicts that the course has with other courses. The total sum of the course's conflicts and the students enrolled in that course is therefore the indicator LWD. The larger the indicator becomes, the more difficult the assignment is. This

method is similar to the one that is used by the director of examinations of U.I when constructing the examination timetable manually. First off, the most difficult exams were assigned to the timetable but afterwards, the easier ones were assigned too.

7.1. Revised Model

By applying this technique some revisions must be done to the constraints originally formulated for Phase I. Hence, the problem will from now on be split into two sub-phases or *Phase I-a* and *Phase I-b*. Phase I-a will schedule the γ most difficult exams into suitable time slots. The solution from Phase I-a will be used in Phase I-b as an initial fixed solution and the rest of the exams will be scheduled into suitable slots along with the initial solution. The models in both phases will be the same except, one new constraint will be added and constraint (5.2) will be slightly revised.

1. **Fix Initial Solution**
   A new constraint must be added to the model such that the solution from Phase I-a is fixed for the γ most difficult courses.

$$x_{c,e} = 1 \qquad \forall c \in \mathcal{C}, \quad \forall e \in F_c^s \tag{7.2}$$

2. **Revised: There Can Be Only One**
   Constraint 5.2 will be revised. Here, the model will only be solved for the γ most difficult courses. Therefore, new boundaries are inserted to distinguish between Phase I-a and Phase I-b. The first condition added is $L_c > \epsilon$, this means that only courses with more than ϵ students assigned are assigned for Phase I-a. Similarly, another condition is made where LWD is added, such that $LWD > \epsilon \cdot p$ where $p \in \{0, 1\}$. Therefore, for Phase I-a $\epsilon = 1$. The condition $M_{c_1,c_2} = 1$ where the courses with questions in common had to be added to guarantee feasibility when splitting the problem into two sub-phases. Similarly, the condition when courses required specific time slots was used for Phase I-a with $|F_c^f| > 0$ with the most difficult courses. The last condition added is when a solution gets fixed from Phase I-a such that $|F_c| > 0$. Hence, constraint 7.3 is added to the model, replacing the original constraint 5.2.

$$\sum_{e \in \mathcal{E}} x_{c,e} = \begin{cases} 1 & S_c > \epsilon \vee LWD > \epsilon \cdot p \vee M_{c_1,c_2} = 1 \\ & \qquad \vee |F_c| > 0 \vee |F_c^f| > 0 \\ 0 & \text{Otherwise} \end{cases} \tag{7.3}$$

7.2. Computational Experiments

In this section, the computational experiments will be performed on the heuristic approach for the revised model as described in Section 7.1. All of the parameters settings will remain the same as described in Chapter 6, but the only difference is that the tolerance for common students having no free day before an exam ϕ was increased from 17 to 20 for Phase I-b but similarly, the tolerance for number of common students having two exams in the same day or in a row had to be increased from 6 to 16. If the tolerances were not increased it led to an infeasible solution since assigning courses to time slots where γ courses had already been assigned was impossible.

Objective	J	Z	$\sum\sum I_{c_1,c_2} A_{c_1,c_2}$	$\sum\sum U_{c_1,c_2} A_{c_1,c_2}$	$\sum\sum P_{c_1,c_2} A_{c_1,c_2}$	Incumbent	Duality Gap	Time [sec]
Phase I-a	128	456	50	37	795	7772,95	0,00%	9568,91
Phase I-b	150	1012	97	69	1324	14623,90	0,00%	8855,35
Total								18424,26

Table 7.1: Computational results for the heuristics approach

7.3. Comparison between methods

In this section, both methods will be compared, the original model solved in a one phase approach and the heuristic model that was solved in a two phase approach as described in the last section. It is not obvious which method has the edge over the other due to various reasons. By inspection, it can be seen that the cost z achieved by the original model is lower than z^* achieved by the revised model. Hence, the revised model gives a result which is a suboptimal solution.

Consequently, the revised model gives a slightly worse result than the original one since $z^* > z$. The main difference is between the students going to exams the same day and students not receiving 24 hours rest. The revised model only gives better results in the same day examinations but the difference between the result is only 12 students while there is a higher difference between the other parts of the objective. Figure 7.1 shows the overall variation between the time slot were the least amount of students are assigned to versus the time slot where the maximum number of students are assigned to over the examination period for both objectives. By looking at Figure 7.1 it seems like the revised model outperforms the original

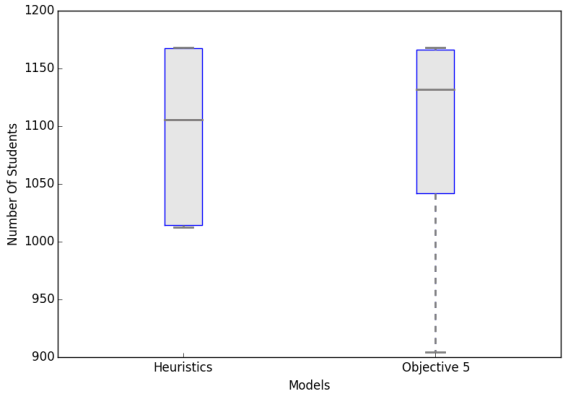

Figure 7.1: Variation between the easiest and the busiest time slot by method

model since there is a lower variation between the easiest and the busiest time slot compared to the original model and therefore, the total number of students assigned to every time slot is similar. Similarly, the revised model outperforms the original model in computational time. The comparisons between methods can be seen in Table 7.2.

Objective	J	γ	$\sum\sum I_{c_1,c_2} A_{c_1,c_2}$	$\sum\sum U_{c_1,c_2} A_{c_1,c_2}$	$\sum\sum P_{c_1,c_2} A_{c_1,c_2}$	Incumbent	Duality Gap	Time [hours]
Heuristic	150	**1012**	**97**	69	1324	14623,90	0,00%	5,12
Objective 5)	**128**	904	109	**52**	**1238**	14865,91	52,30%	60,00

Table 7.2: Comparison of computational results between methods

The table shows that it depends on what counts as a satisfying solution for the U.I and how much time is available to make the examination scheduling. If there is enough time to make the timetable, the original model should be used with objective 5. However, as the situation is now the data changes on a daily basis. Therefore, it would be better to use the revised model since there is a time limitation to the problem.

.

7.4. Summary

In this section, a heuristic approach was designed by using the largest weighted degree which is a low-level heuristic as proposed by Carter et al. (1996). This step was done since solving the problem in a one phase approach was computationally expensive. As the heuristic was applied, one new constraint had to be added and one constraint had to be revised from the original model. Phase I was now partitioned into two sub-phases or Phase I-a which will schedule the γ most difficult exams and then Phase I-b which uses the solution from Phase I-a and schedules the rest of the exams. The heuristics approach was solvable in approximately 5,12 hours whereas the one phase approach left a duality gap $52,30\%$ after $60,00$ hours. By splitting the problem into a two phase approach only leads to a sub-optimal solution that could not outperform a one phase approach. However, the heuristic solution is not considered a bad solution. It is less computationally expensive and only gives slightly worse results over all. In a perfect world where there are no time limitations, a one phase approach should be used but since the data changes on a daily basis, the heuristic method is a more reasonable choice for the problem.

8. Discussions

The results achieved by using a one phase approach for the examination scheduling had a very long computational time. As a consequence, a heuristic was proposed where the most difficult courses were scheduled in the first phase and then the remaining courses were scheduled after that in the second phase. The two phase approach was solvable in a reasonable time while the one phase approach was not. Splitting the problem into two sub-problems is not considered to be a good approach although it has been done in the literature before. Doing so could lead to infeasibility of the model for later assignments of other exams (Qu et al. 2009). Similarly, using heuristics in general only give a suboptimal solution.

It can be seen in Table 7.2 in Chapter 7 that the two phase heuristic did not give a better solution than the 60 hours run. Although the solution achieved by the heuristics is not bad, it is not as good as the one phase approach. By using the heuristic means lower quality solutions for the students. However, the difference between the solutions does not vary that much. Therefore, splitting Phase I into two sub phases was considered to be acceptable this time. One may ask why this problem is so computationally expensive and why it had to be split into two sub-phases. In general, those type of problems have proven to be NP-Complete and therefore really hard to solve. Similarly, by looking at Figure 4.3, it can be seen that there is a high correlation between almost all courses of the university. If students were only to follow their curriculum and there would be no elective courses at all, the problem would be easier to solve. However, this is not the case since most of the students are permitted to choose freely. This fact along with all of the conditions that must be satisfied for the problem makes the problem very computationally expensive.

It is interesting to see how many requirements arrived from the departments and schools regarding the exams. When such requirements are included the search space decreases for a potential solution that will make the problem less computationally expensive. Hence, it is essential to have some sort of requirements to decrease the search space but having too many will damage the solution for the students. In this study, there were 130 requirements, meaning that roughly 25% of the exams that were scheduled had a required time slot/s.

In the model, forced conditions were allowed to violate the hard constraints in some cases. After all, that was not the best way since two large courses on the first year

in the School of Business had day after day examinations in the official solution for the U.I this autumn. In that occasion, the teachers of the courses asked for two consecutive dates that was put in the requirement file causing this to happen. Although, the solution was reviewed under time pressure by the heads of School of Business and the Director of Examinations it was still overlooked. Hence, some process changes must be made to avoid situations like that from happening but inspection by eye is very hard when the exams are that many.

The room scheduling model performed well but some time slots were more difficult to schedule than others. The main problem arises when two or more large exams are assigned to the same time slot. This could easily happen since violation of the hard constraints 5.9 was allowed for forced conditions. As mentioned in Section 6.1 a large course was indicated as a course with more than 260 students assigned. By looking at Figure 6.6 it can be seen that violation of this constraint occurs several times. In time slot 3, three courses with more than 250 students were assigned to that slot. Such conditions can lead to problems when scheduling the exams into rooms since the courses might need to be spread into many rooms and buildings due to limitations of rooms with large capacity. In this study, courses with questions in common are treated as individual courses. Although Constraint 5.38 is added that tries to guarantee that the courses with questions in common are assigned to the same buildings, it may not work as expected. Therefore, I think from the start, such courses should be combined into one large course rather than few individuals. As the problem is formulated now, circumstances can happen where courses with questions in common that vary in sizes but have the sum over 260 students can be assigned along with one large course and perhaps in addition to some forced conditions of other large courses. This could lead to instances where many large courses can be assigned to one time slot but still not violate constraint 5.9.

It was really interesting working on models on the go and almost making the first advanced testing on the university itself. A lesson learned for the author and the team is that it would have been better to start with a smaller sub-problem rather than the whole university e.g. one school or to dig deeper into investigations on the behavior of the models and to get a good understanding of the solution beforehand. As the problem expands, the harder it is to investigate every single course in every single department and school. If it would have been done, I guess some of the complaints could have been avoided. However, it was a very valuable opportunity to do the testing on the university itself and it probably took the project a lot further. There were many meetings in a short period of time and frequent changes were made on the models. It was discovered that the problem had to be split into two sub-problems for the scheduling, since it took a shorter amount of time to achieve a solution that was necessary due to daily changes of the data. For Phase II, more advanced testing on the model and revisions of the solutions were performed beforehand and therefore it performed better than Phase I.

9. Conclusions

In this thesis, two mixed integer programming models were proposed for the examination timetabling problem at the University of Iceland where Phase I schedules the exams to suitable time slots while Phase II schedules the exams to rooms. Before, the problem had only been solved manually by the director of examinations requiring $5 - 7$ days each semester. Making such models was a breakthrough in this process where the goal was first and foremost to minimize the time spent in timetabling and potentially give more quality solutions for the students.

The main experiments performed for Phase I was to investigate the most suitable objective for the model. Hence, five different combinations of objectives were investigated. As more elements were used for the objective, the problem becomes more computationally expensive. However, the computational time is not only bound with the objective but also with the problem that each course at the U.I is connected with almost every other course due to the fact that the students are permitted to choose courses almost freely making the scheduling very difficult to solve. In order to choose the right objective for the model, all of the five objectives were tested with termination time of three hours. Afterwards, more advanced tests were made with termination time of 60 hours for the three most promising objectives. None of the most promising objectives could finish the run during that time, but the best duality gap achieved was $47,02\%$ for objective 4. However, by an investigation, objective 5 was chosen best since it gave the best overall results. The best objective consisted of minimization of the clashes in the same time slot, same day examinations for students and students not receiving one day off before an exam. Similarly, it included that the computer exams to be spread as evenly as possible throughout the examination period and that the total number of assignments to each time slot should be as equal as possible.

Since Phase I was very computationally expensive, a simple low level heuristic method was proposed similar to Carter et al. (1996) used in their work was used to see if it was possible to make the problem less computationally expensive and achieve a good solution using such technique. Hence, the problem was split into two sub-problems or Phase I-a and Phase I-b. Phase I-a scheduled the γ most difficult exams into time slots while Phase I-b scheduled the remaining exams along with the difficult ones. The results achieved, gave a suboptimal solution and could not outperform a solution using only a one phase approach. However, the difference

between the solutions does not vary that much. The main difference was the examination schedule was now solvable in a reasonable time using a two sub-phases approach (5, 12 hours) but not in one phase approach (60 hours). As the situation is now at the University of Iceland, there are no deadlines of submission of requirements exists and therefore it would not make sense to use a one phase approach due to rapid changes of the data until shortly before publication of the examination scheduling.

The main computational experiments performed for Phase II was to see how good the room efficiency was when a room was used and how often an exam could be assigned to it's preferred or required buildings. The model performed well and gave a very good room efficiency in general and managed to assign almost all exams to their preferred or required buildings. However, what makes the room scheduling difficult is when two or more large courses are assigned to the same time slot due to limitations of rooms with large capacity. That situation can happen easily since violation the of the constraint which guarantees situations like that not to happen is allowed to some degree (*e.g. forced requirements*) and that courses with questions in common are treated as several courses rather than one. Such occasions could also violate that constraint in an indirect way and hence, two or more large courses could be assigned to the same slot.

In conclusion, the results of using models making the examination scheduling and the room scheduling for the University of Iceland gave promising results and could potentially make the process easier for the director of examinations. However, further improvements of the models and the processes are necessary.

9.1. Future Work

As has previously been discussed in this thesis, the problem is very computationally expensive by using a one phase approach for Phase I. Therefore, a simple low level heuristics was proposed to make the problem solvable in a reasonable time. The results of using a heuristic approach instead of a one phase approach only gave a slightly worse result but was less computationally expensive. That step was necessary since the data changed daily the days before the publication of the examination table and therefore, the solutions had to be generated quickly. However, such heuristics approaches are not recommended in Qu et al. (2009) survey since they can generate infeasible solutions. Therefore, a new approach must be applied to make the problem less computationally expensive.

What I believe could make the problem less computationally expensive, is to make a curriculum based scheduling rather than post enrollment based scheduling as pro-

posed in this thesis. By only looking at each individual curriculum, the problem should get easier to solve since most of the students follow curricula. Therefore, the priority will be put on the curricula to satisfy each goal rather than each individual exam. By doing this, each curriculum should not be having two exams in the same day, within 24 hours and will at minimum receive one day off before each exam. The main drawback is that the students are permitted to choose courses freely between schools and departments. This means that each individual student in the model will not count as much as curriculum. Therefore, each individual student can now end up in same day examinations if it does not belong to curricula. The question will arise on fairness and what is more satisfactory for the university. One of the main problem was if you belong to a small curriculum, that group could end up in daily examinations since the tolerances ϕ and α that allowed some number of students to have no day off before an exam, examinations within 24 hour or same day examinations were bigger than those following the curricula. By using curriculum based scheduling these kind of circumstances would not happen. However, many students are taking courses that belong to different educational years and do not study full time. The university is not collecting such information directly and therefore the question arises on how such curricula should be constructed. Some sort of clustering technique could be used, finding these groups. This could be interesting to take a look at in another research.

The main drawback of the model proposed for Phase II was the courses with questions in common could end up in many rooms. This could be really inconvenient for the teachers of the courses that would need to visit many rooms. Therefore as future work, I believe it is better to treat courses with questions in common as just one course rather than few individual courses. Other future work, is to model the invigilation staff scheduling and the special student scheduling. The special student scheduling is very difficult to model and does not follow the same scheduling rules as for the general students since it needs to be individual based rather than course based.

References

S. M. Al-Yakoob, H. D. Sherali, and M. Al-Jazzaf. A mixed-integer mathematical modeling approach to exam timetabling. *Computational Management Science*, 7 (1):19–46, 2007.

P. Amaral and T. C. Pais. Compromise ratio with weighting functions in a tabu search multi-criteria approach to examination timetabling. *Comput. Oper. Res.*, 72(C):160–174, Aug. 2016. ISSN 0305-0548.

T. Arani and V. Lotfi. A three phased approach to final exam scheduling. *IIE Transactions*, 21(1):86–96, 1989.

T. Arbaoui, J.-P. Boufflet, and A. Moukrim. Preprocessing and an improved mip model for examination timetabling. *Annals of Operations Research*, 229(1):19–40, 2015.

A. Bettinelli, V. Cacchiani, R. Roberti, and P. Toth. An overview of curriculum-based course timetabling. *TOP*, 23(2):313–349, 2015.

B. Bilgin, E. Özcan, and E. E. Korkmaz. An experimental study on hyper-heuristics and exam timetabling. In E. K. Burke and H. Rudová, editors, *Practice and Theory of Automated Timetabling VI: 6th International Conference, PATAT 2006 Brno, Czech Republic, August 30–September 1, 2006 Revised Selected Papers*, pages 394–412, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.

A. Bonutti, F. De Cesco, L. Di Gaspero, and A. Schaerf. Benchmarking curriculum-based course timetabling: formulations, data formats, instances, validation, visualization, and results. *Annals of Operations Research*, 194(1):59–70, 2012.

S. Broder. Final examination scheduling. *Commun. ACM*, 7(8):494–498, Aug. 1964.

E. Burke, D. Elliman, P. Ford, and R. Weare. Examination timetabling in british universities: A survey. In E. Burke and P. Ross, editors, *Practice and Theory of Automated Timetabling: First International Conference Edinburgh, U.K., August 29–September 1, 1995 Selected Papers*, pages 76–90, Berlin, Heidelberg, 1996. Springer Berlin Heidelberg.

E. K. Burke and S. Petrovic. Recent research directions in automated timetabling. *European Journal of Operational Research*, 140(2):266 – 280, 2002.

E. K. Burke, G. Kendall, M. Misir, E. Özcan, E. K. Burke, G. Kendall, E. Özcan, G. Kendall, and M. Misir. Applications to timetabling. In *Handbook of Graph Theory, chapter 5.6*, pages 445–474. Chapman Hall/CRC Press, 2004.

E. K. Burke, B. McCollum, A. Meisels, S. Petrovic, and R. Qu. A graph-based hyper-heuristic for educational timetabling problems. *European Journal of Operational Research*, 176(1):177 – 192, 2007.

M. W. Carter. A survey of practical applications of examination timetabling algorithms. *Oper. Res.*, 34(2):193–202, Mar. 1986.

M. W. Carter and G. Laporte. Recent developments in practical examination timetabling. In *Selected Papers from the First International Conference on Practice and Theory of Automated Timetabling*, pages 3–21, London, UK, UK, 1996. Springer-Verlag.

M. W. Carter, G. Laporte, and S. Y. Lee. Examination timetabling: Algorithmic strategies and applications. *The Journal of the Operational Research Society*, 47 (3):373–383, 1996.

A. Cataldo, J.-C. Ferrer, J. Miranda, P. A. Rey, and A. Sauré. An integer programming approach to curriculum-based examination timetabling. *Annals of Operations Research*, pages 1–25, 2016. URL `http://dx.doi.org/10.1007/s10479-016-2321-2`.

A. J. Cole. The preparation of examination time-tables using a small-store computer. *The Computer Journal*, 7(2):117–121, 1964.

T. B. Cooper and J. H. Kingston. The complexity of timetable construction problems. In *Selected Papers from the First International Conference on Practice and Theory of Automated Timetabling*, pages 283–295, London, UK, UK, 1996. Springer-Verlag. ISBN 3-540-61794-9.

A. Dammak, A. Elloumi, and H. Kamoun. Classroom assignment for exam timetabling. *Advances in Engineering Software*, 37(10):659 – 666, 2006. ISSN 0965-9978. doi: http://dx.doi.org/10.1016/j.advengsoft.2006.02.001.

M. Dimopoulou and P. Miliotis. Implementation of a university course and examination timetabling system. *European Journal of Operational Research*, 130(1):202 – 213, 2001.

M. Eley. Ant algorithms for the exam timetabling problem. In E. K. Burke and H. Rudová, editors, *Practice and Theory of Automated Timetabling VI: 6th International Conference, PATAT 2006 Brno, Czech Republic, August 30–September 1, 2006 Revised Selected Papers*, pages 364–382, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg. doi: 10.1007/978-3-540-77345-0_23.

D. Johnson. Timetabling university examinations. *The Journal of the Operational Research Society*, 41(1):39–47, 1990.

M. Kahar and G. Kendall. The examination timetabling problem at universiti malaysia pahang: Comparison of a constructive heuristic with an existing software solution. *European Journal of Operational Research*, 207(2):557 – 565, 2010.

B. A. Kassa and A. E. Tizazu. Personnel scheduling using an integer programming model- an application at avanti blue-nile hotels. *SpringerPlus*, 2(1):333, 2013.

G. Laporte and S. Desroches. Examination timetabling by computer. *Computers Operations Research*, 11(4):351 – 360, 1984.

T.-Y. Leong and W.-Y. Yeong. A hierarchical decision support system for university examination scheduling. Technical report, working paper, National University of Singapore, 1990.

R. Lewis, B. Paechter, and B. Mccollum. Post enrolment based course timetabling: A description of the problem model used for track two of the second international timetabling competition, 2007.

B. McCollum, A. Schaerf, B. Paechter, P. McMullan, R. Lewis, A. J. Parkes, L. D. Gaspero, R. Qu, and E. K. Burke. Setting the research agenda in automated timetabling: The second international timetabling competition. *INFORMS Journal on Computing*, 22(1):120–130, 2010.

B. McCollum, P. McMullan, A. J. Parkes, E. K. Burke, and R. Qu. A new model for automated examination timetabling. *Annals of Operations Research*, 194(1): 291–315, 2012.

Metaheuristics Network. International timetabling competition, 2002. URL `http://sferics.idsia.ch/Files/ttcomp2002/`.

T. Müller. Itc2007 solver description: a hybrid approach. *Annals of Operations Research*, 172(1):429, 2009.

T. Müller. Real-life examination timetabling. *Journal of Scheduling*, 19(3):257–270, 2016.

T. Müller and H. Rudová. Real-life curriculum-based timetabling with elective courses and course sections. *Annals of Operations Research*, 239(1):153–170, 2016.

N. Pillay and W. Banzhaf. A study of heuristic combinations for hyper-heuristic systems for the uncapacitated examination timetabling problem. *European Journal of Operational Research*, 197(2):482 – 491, 2009.

G. Post, L. Di Gaspero, J. H. Kingston, B. McCollum, and A. Schaerf. The third international timetabling competition. *Annals of Operations Research*, 239(1): 69–75, 2016.

R. Qu, E. K. Burke, B. Mccollum, L. T. Merlot, and S. Y. Lee. A survey of search methodologies and automated system development for examination timetabling. *J. of Scheduling*, 12(1):55–89, Feb. 2009.

A. Schaerf. A survey of automated timetabling. *Artif. Intell. Rev.*, 13(2):87–127, Apr. 1999.

G. Schmidt and T. Ströhlein. Timetable construction – an annotated bibliography. *The Computer Journal*, 23(4):307–316, 1980.

E. H. T. Birbas, S. Daskalaki. Timetabling for greek high schools. *The Journal of the Operational Research Society*, 48(12):1191–1200, 1997.

University Of Iceland. Starfsmenn, 2016. URL `http://www.hi.is/adalvefur/starfsmenn`.

University Of Iceland. Heildarskráning nemenda í Háskóla Íslands frá upphafi, 2017. URL `http://www.hi.is/adalvefur/heildarskraning_nemenda_i_haskola_islands_fra_upphafi_0`.

R. C. Vahid Lotfi. A final-exam-scheduling package. *The Journal of the Operational Research Society*, 42(3):205–216, 1991.

D. C. Wood. A system for computing university examination timetables. *The Computer Journal*, 11(1):41–47, 1968.

A. Required and Preferred Buildings

All departments within each school have their own home-building where their examinations should be assigned if possible in order to have the students as familiar with the housing. In Table A.1 all of the home buildings are listed for each department where the examinations should preferably be assigned. However this is not always a possibility since some big courses might need to be assigned elsewhere on the university area.

Therefore, there exist another list with buildings which are used as a second option where the examinations of the department should be assigned if it cannot be assigned to their home building. The second option buildings are usually a list of all of the buildings at the main university central area with slight variations between departments. In Table A.2 all of the second option buildings are listed for each department in an increasing priority order.

Building	Departments
Adalbygging	Matvæla- og næringarfræðideild
Askja	Jarðvísindadeild Líf- og umhverfisvísindadeild
Arnagardur	Deild erlendra tungumála, bókmennta og málvísinda Guðfræði- og trúarbragðafræðideild Íslensku- og menningardeild Sagnfræði og heimspekideild
Eirberg	Hjúkrunarfræðideild Læknadeild Tannlæknadeild
Hamar	Íþrótta-, tómstunda og þroskaþjálfaradeild Kennaradeild Uppeldis- og menntunarfræðideild
Haskolatorg	Félags- og mannvísindadeild Félagsráðgjafardeild Hagfræðideild Lagadeild Stjórnmálafræðideild Viðskiptafræðideild
VR-II	Lyfjafræðideild Iðnaðarverkfræði-, vélaverkfræði- og tölvunarfræðideild Rafmagns- og tölvuverkfræðideild Raunvísindadeild Umhverfis- og byggingarverkfræði

Table A.1: Preferred buildings by departments

Departments	Set of buildings
Félags- og mannvísindadeild Hagfræðideild Lagadeild Stjórnmálafræðideild Deild erlendra tungumála Guðfræði- og trúarbragðafræðideild Íslensku- og menningardeild Sagnfræði og heimspekideild	Haskolatorg, Oddi, Logberg, Gimli, Arnagardur, Nyi-Gardur
Viðskiptafræðideild Lyfjafræðideild Umhverfis- og byggingarverkfræði	VR-II, Haskolatorg, Oddi, Logberg, Gimli, Arnagardur, Nyi-Gardur
Hjúkrunarfræðideild Læknadeild Tannlæknadeild Félagsráðgjafadeild	Eirberg, Haskolatorg, Oddi, Logberg, Gimli, Arnagardur, Nyi-Gardur
Matvæla- og næringarfræðideild	Adalbygging, Haskolatorg, Oddi, Logberg, Gimli, Arnagardur, Nyi-Gardur
Sálfræðideild	Haskolatorg, Oddi, Logberg, Gimli, Arnagardur, Nyi-Gardur, Hamar
Íþrótta-, tómstunda og þroskaþjálfaradeild Kennaradeild Uppeldis- og menntunarfræðideild	Hamar, Haskolatorg, Oddi, Logberg, Gimli, Arnagardur, Nyi-Gardur, Klettur, Enni
Iðnaðarverkfræði-, vélaverkfræði- og tölvunarfræðideild Rafmagns- og tölvuverkfræðideild	VR-II, Haskolatorg, Oddi, Logberg, Gimli, Arnagardur, Nyi-Gardur, Askja
Jarðvísindadeild Líf- og umhverfisvísindadeild	Askja, Haskolatorg, Oddi, Logberg, Gimli, Arnagardur, Nyi-Gardur, VR-II
Raunvísindadeild	VR-II, Haskolatorg, Oddi, Logberg, Gimli, Arnagardur, Nyi-Gardur Eirberg

Table A.2: Required buildings for departments

B. Bessý - The Examination Timetabling System

Bessý is a examination timetabling system for the University of Iceland. The system itself consists of two parts, Part I which is the examination scheduling itself (*assignment of courses to slots*) and Part II which is the room scheduling using the solution from Part I. The system itself consist of multiple codes, commands and transformation of the data to the right format. Hence, in this handbook of usage a brief description on every element of the system will be discussed briefly. The handbook will be divided into following parts. Firstly, all of the main data used will be presented in Section B.1. Secondly, the system maps will be presented in Section B.2 and all of its detail described.

B.1. Data files

In this section all of the data files that are used for the models are described and how they are transformed to the right format for the model.

B.1.1. Courses.dat

As can be seen on figure B.1, two data files are needed. The first file is *Students.csv* which contains the students registrations to all of the courses of UI as described in section 3.3.2. Since this file contains all of the registrations of students, the courses that do not require any examinations must be filtered out. This is done by supplying the data *Courses.txt* which includes all of the courses require examinations. This list is made from *Courses.csv* and should indicate the courses that should be scheduled. The output of *Clashes.m* is the file *Courses.dat* which consists of the exams that should be examined, the total number of registrations to each course and similarly the number of common students between courses. All of the original files consisted of Icelandic letters but in the output they have been transposed to English alphabetic letters. The flow for this process can be seen on Figure B.1 along with the example

of the input and output.

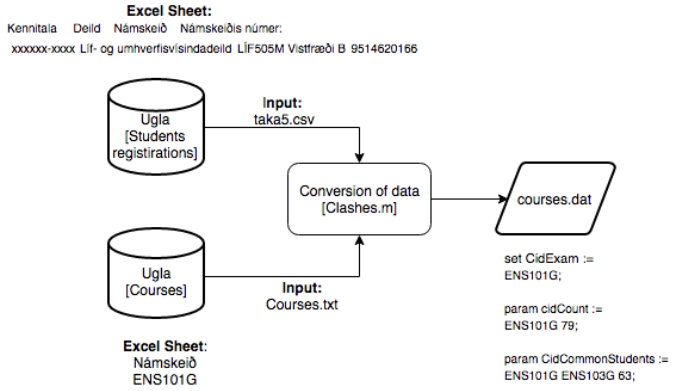

Figure B.1: The data import, transformation and export for the code Clashes.m

B.1.2. Forsendur.dat

In figure B.2, the data files the *conjoined.csv* and *Requirements.csv* are imported for the code. The data *conjoined.csv* includes all of the lectures that are taught as overlapped and therefore should be examined at the same time as was discussed in Section 3.3.1. At the same time, another file is imported *Requirements.csv* which includes all of the requirements or wishes from departments, schools or the examination director as described in Section 3.3.1. The main purpose of the usage of this code is to read the files and transform them on the right format for the model. Example of the input data and the output *forsendur.dat*, can be seen in the figure.

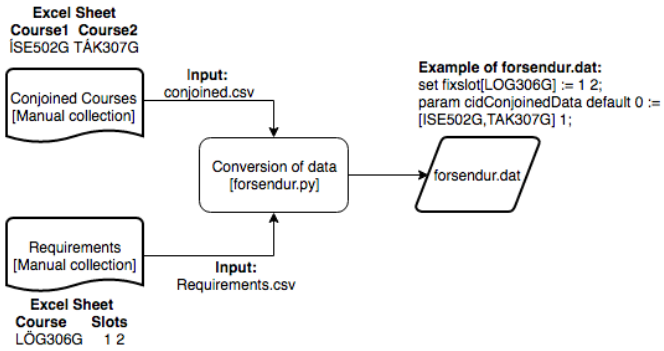

Figure B.2: The flow for the code forsendur.py

B.1.3. Resources.dat

In Figure B.3 the file *Courses.csv* is imported to the code which was discussed in 3.3.2 in the last paragraph of that section. The main purpose of this code is to generate a new file, *resources.dat* that contains a list of the courses that have students with disabilities, and a list with the total the number of the students with the disabilities assigned to each course. At the same time a list of courses that require computers for their examinations and courses that should not be assigned to any rooms can similarly be found in this file. An overview of the flow along with the example of the input and the output can be seen on Figure B.3.

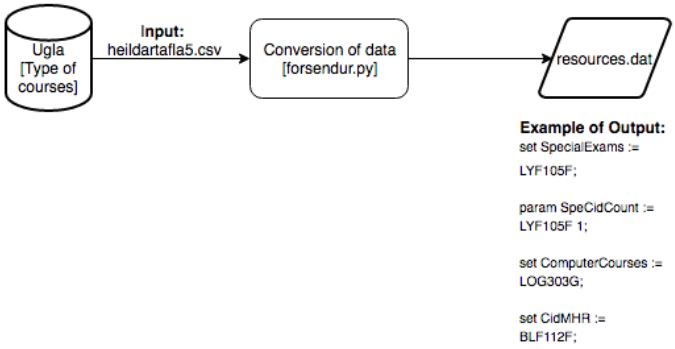

Figure B.3: The data import, transformation and export for the code Resources.py

B.1.4. Default.dat

As can be seen on Figure B.4 the file *RoomAssignment.csv* is imported to the code since it includes the departments and school numbers for each course as was discussed in Section 3.3.2. The main purpose of this code is to make lists of courses that should be assigned for Stakkahlíð, but most of the courses that need to be assigned there belong to the School of Education. The flow for the the code can be seen in Figure B.4 with examples of the input and the output.

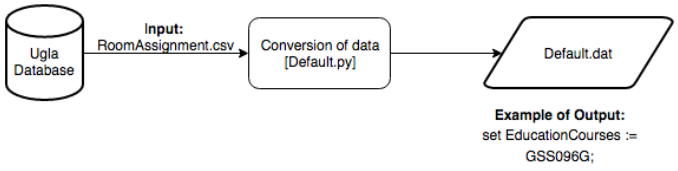

Figure B.4: The data import, transformation and export of the code Default.py

B.1.5. PriorityBuildings.dat

As mentioned in Section 3.3.1 does department have a preferred where their examinations should be assigned to. The list of home buildings for each school can be seen in Appendix A in Table A.1. The main purpose of *PriorityBuildings.py* is to read the data from *RoomAssignment.csv* where department numbers can be found for each course as discussed in the last paragraph in Section 3.3.2. The main purpose of the algorithm is to match the preferred building by department number to the courses within each department. The flow of this step can be seen in Figure B.5 with examples of the input and the output.

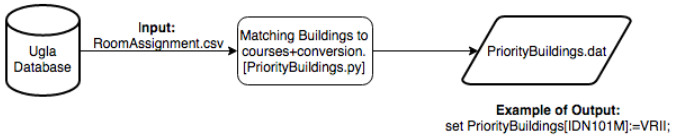

Figure B.5: The flow of the code PriorityBuildings.py

B.1.6. RequiredBuildings.dat

The main purpose of this step is similar to the *PriorityBuildings.py* as presented in Section B.1.5 but the main difference is that instead of only linking department number of a course to its most preferred building now it is linked to second option buildings which are suitable for examinations for the department as described in Section 3.3.1 and can be seen in Table A.2. Each course is now linked to multiple buildings correlated to its department number rather than one building as previously done. The flow of the code can be seen in Figure B.6 with examples of the input and the output.

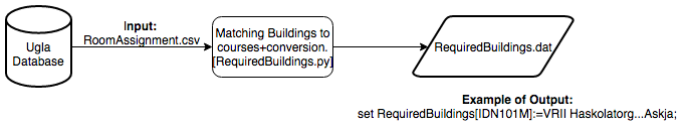

Figure B.6: The flow of the code RequiredBuildings.py

B.1.7. RoomData.dat

In Figure B.7, can the step of transforming the file *Rooms.csv* to the right format that can be used with the model. The file contains information about each room regarding in which building it is located in, the total number of seats, the type of the room and which room priority coefficient it receives as discussed in Section 3.3.1. The flow of this step can be seen on the figure along with the example of the input and the output.

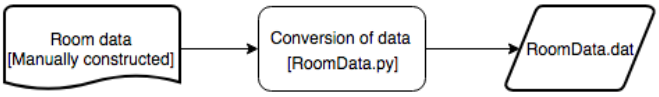

Figure B.7: The flow of the code RoomData.py

B.2. System Maps

In this section, the overall system maps will proposed. The system is in two Parts, Part I which includes all steps needed for the examination scheduling and Part II which includes all steps needed for the room scheduling by using the solution from Part II. This section will be divided in two parts Part I and Part II. Each part will firstly present a overall systematic map with all steps numbered. Afterwards, each step will be discussed briefly. The main data used for both parts was presented in Section B.1.

B.2.1. Part I

In this section a system map will be presented for Part I of the system which consist of all steps regarding the examination scheduling itself. As can be seen in Figure B.8 the system map for the part I can be seen. It consists of 12 steps which will be introduced more closely to the reader later on in this section in a numbered order. The first four steps have previously been presented in Section B.1 and won't be for further coverage.

5. Examination Scheduling model
In this part the formulation of the model for the examination scheduling itself as described in chapter 3.2 takes place. When the model is ready it will be used for part 6 as the file model file *Bessy.mod*. The model will be written in GLPK (Gnu Linear Programming).

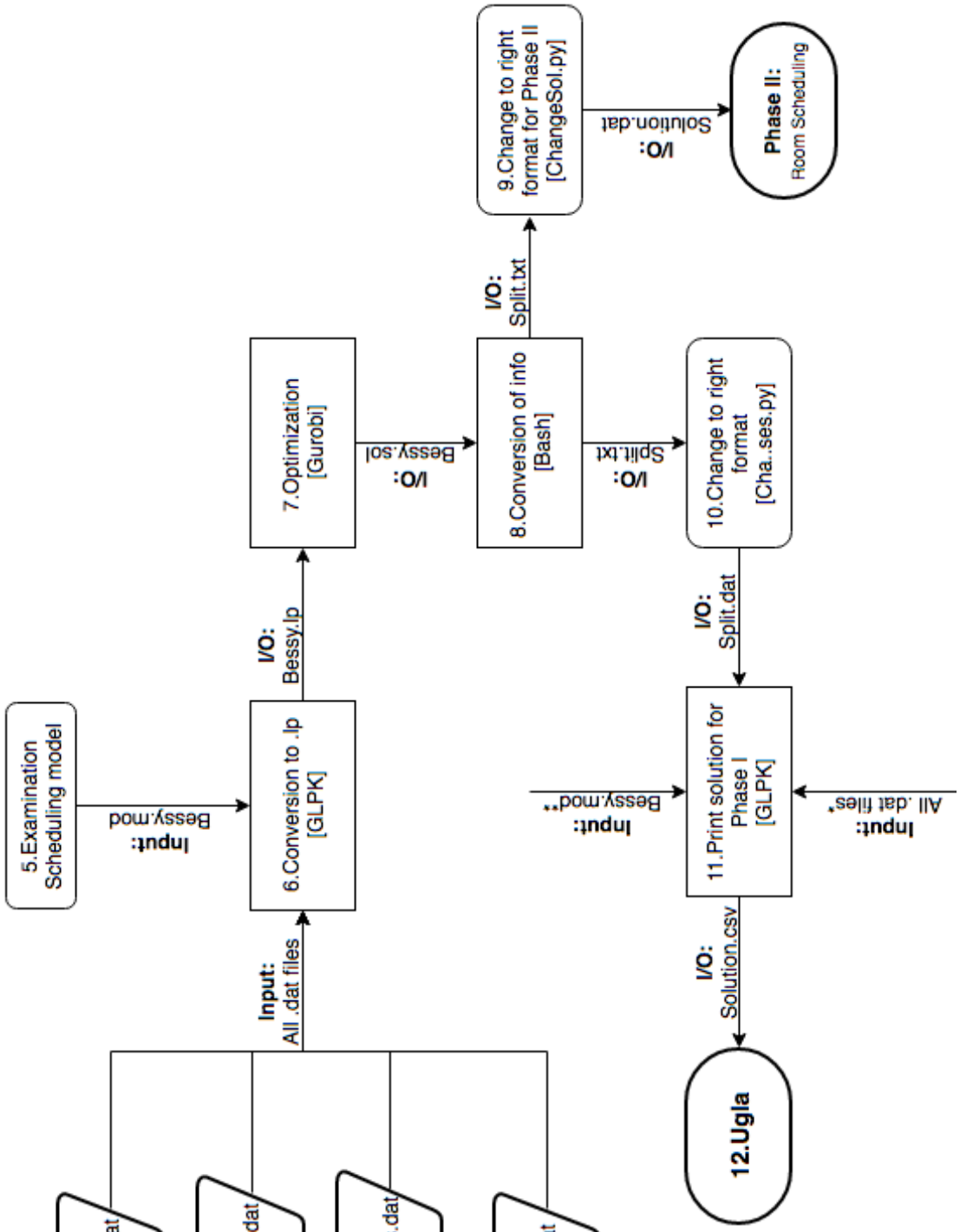

Figure B.8: The system map for Part I which represents Phase I and all of its parts[1]

The * indicator that can be found on the figure corresponds to the data marked by the numbers 1, 2, 3 and 4.

6. Conversion to .lp

In this part the conversion of the model and the data files into a *lp* file will be done. This is done since the Gurobi will be used as a solver and it needs a *lp* file to work. In general, a *lp* file writes out every possible combination of each constraint and the objective function. Then the solver will try to find the optimal solution for the problem. The flow of this part can be seen in Figure B.9 along with the bash command that is needed. The solver GLPSOL will be used to convert the model and the data into a *lp* file. The output will be *Bessy.lp*.

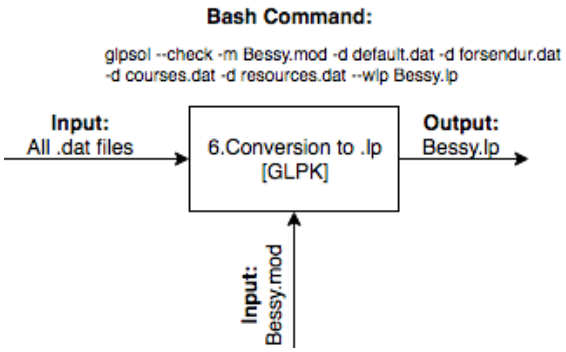

Figure B.9: Conversion of the data and the model to a lp file for Phase I

7. Optimization

In this part the optimization takes place using the solver Gurobi as previously mentioned. The input file will be *Bessy.lp* which was made in step 6. After the the optimal solution has been found it, it is written to the file *Bessy.sol*. The flow of this part can be seen in Figure B.10 along with the bash command needed to run the optimization.

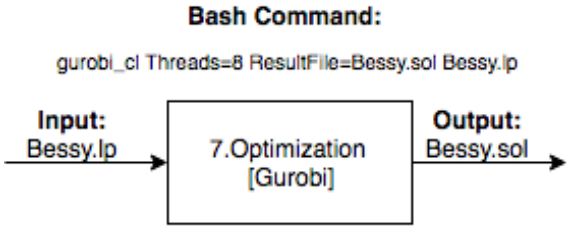

Figure B.10: The optimization for Phase I

8. Conversion of info

Since the solution file *Bessy.sol* contains both zeros and non-zeros elements, the zeros elements must be filtered away from data since they are not needed for further processing as can be seen in Figure B.11. The output will be *Split.txt* which will only contain the solution for the examination scheduling itself. In Figure B.11, the flow of the part can be seen along with the bash command needed for the conversion of the info.

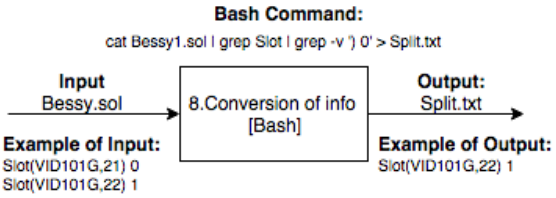

Figure B.11: Conversion of the sol to a txt with filtered information

9. Change to Right format for Phase II

In this part the solution of the examination scheduling will be transformed to the right format so it can be used for Phase II where the room scheduling takes place. The flow of the part can be seen in Figure B.12 and the example of the input and the output. The python code *ChangeSlot.py* will be used to covert the the input file *Split.txt* to the output file *Solution.dat*.

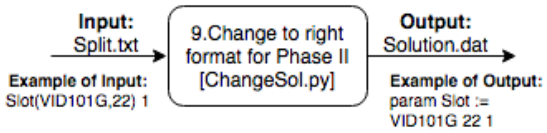

Figure B.12: Transformation of the data to the right format for Phase II

10. Change to right format

Similarly to part 9, this part is used to convert the solution of the examination scheduling and transform it to the right format and used in next step. Here, the code *ChangeToRightFormatForPhases.py* is used to transform the solution from the file *Split.txt* and writes the transformation to the file *Split.dat* as can be seen in Figure B.13.

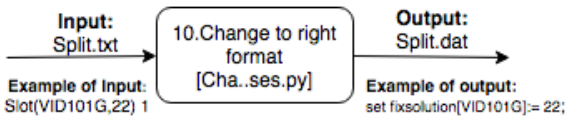

Figure B.13: Transformation of the data to the right format

11. Print solution for Phase I

This part is similar to part 6 where the model and the data was converted to an LP file using the GLPSOL solver. In this part the only difference is that now the GLPSOL solver is used to print the solution nicely. Here, the output is the file *Solution.csv* containing the courses and their time slot assignments and here is also a chance of printing additional information about the solution. The flow of this part can be found in Figure B.14 along with the Bash command needed to perform this procedure.

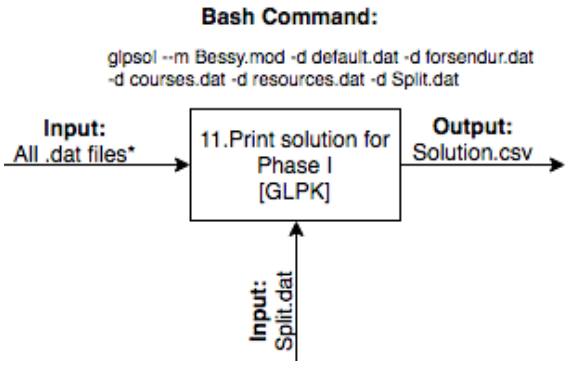

Figure B.14: The print phase of the solution for the examination scheduling

12. Ugla

This is the final step before the a solution is imported to the UGLA examination timetabling visualizer which the director of examinations uses while constructing the examination scheduling and the room scheduling. The data that will be read into the system is on a *csv* format where the courses and information about which time slot they should be assigned to can be found. After it has been imported, the solution can be visualized in the system.

Figure B.15: Examination Scheduling export to UGLA

B.2.2. Part II

In this section Part II of the system will be discussed but Part II consist of all steps regarding the room scheduling. The system map of Part II can be seen in Figure B.17. The system consists of 15 steps and each step will be described in detail later in this section. However the first seven steps, have already been discussed in Section B.1 and step 8 in Section B.2.1.

8. Solution from Phase I
As described in Section B.2.1 then the output of part 9 is the examination scheduling itself converted to the right format that can be used for Phase II. So here, this data will be imported after a solution has been generated for the examination scheduling and the room scheduling takes place. The example of this can be seen in Figure B.12.

9. Solve certain slot
In Section 3.2 was chosen to solve the room scheduling for each slot one by one rather than solving all of the slots at the same time since that is the way it is solved by the director of examinations. This means, that the data *SolveSlot.dat*, must be changed every single time. In figure B.16 can the bash commands needed to change the the data be seen. First, in order to make the the file the initial bash command must be entered for time slot 1. In every single run after the change slots bash command must be entered but the number must be changed for the right slot.

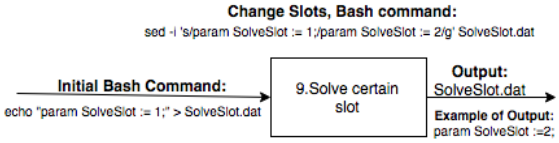

Figure B.16: Solve a certain time slot

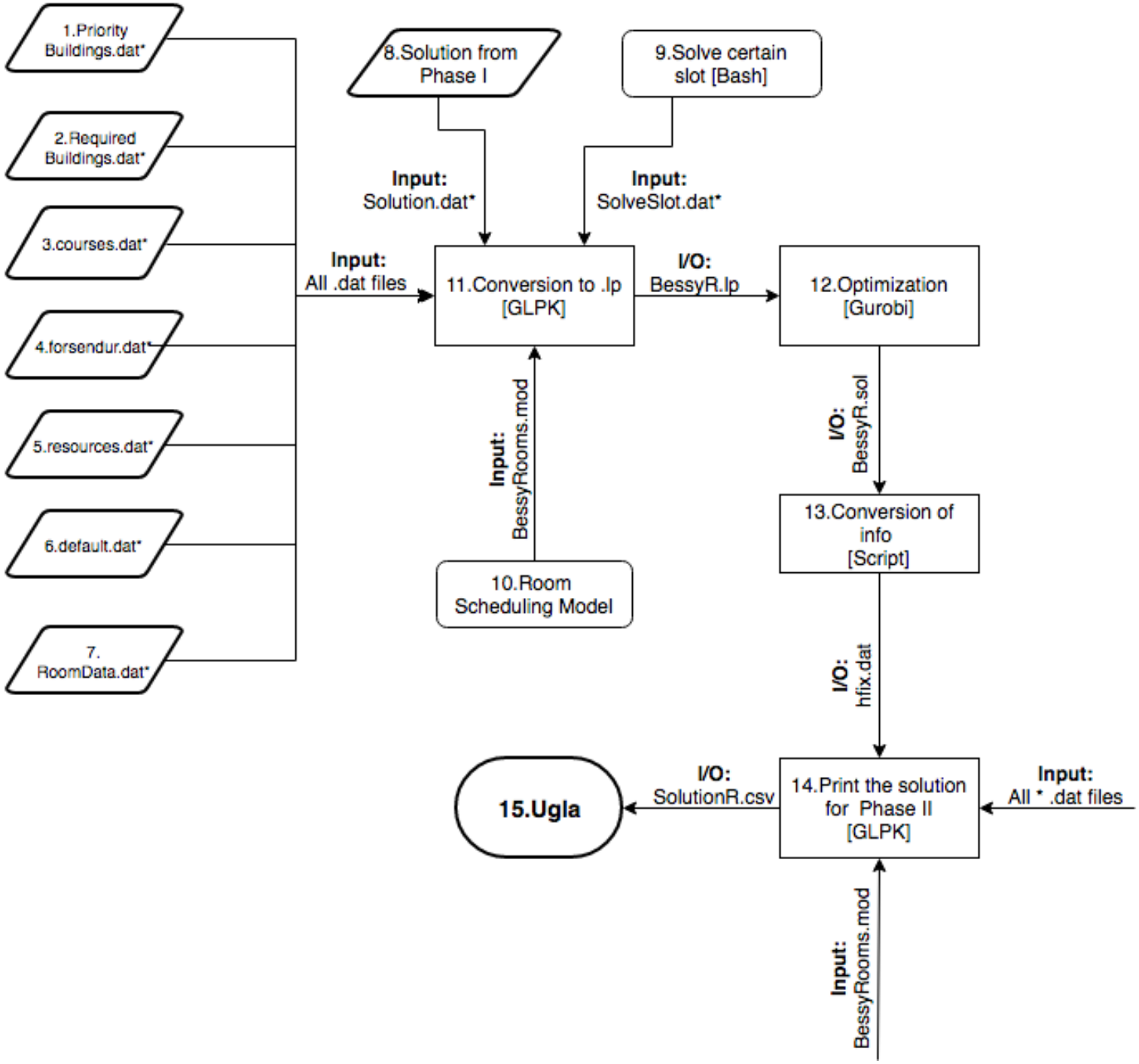

Figure B.17: The system for the room scheduling and all of its parts

10. Room Scheduling Model
In this part the model will be formulated meeting all of the requirements and goals as described in Section 3.2. When the model is ready it will be used as an input to part 11 as the file *BessyRooms.mod*. The model will be written in GLPK.

11. Conversion to .lp
In this part, the conversion of data files and the model file to an *lp* file takes place by using the solver GLPSOL. In general, *lp* files consist of all combinations of the constraints and the objective according to the data. This step is necesary since in next part, the solver Gurobi will be used and in order to use it a *lp* must be imported as can be seen in Figure B.18. The output will *BessyR.lp* which can be used for the optimization.

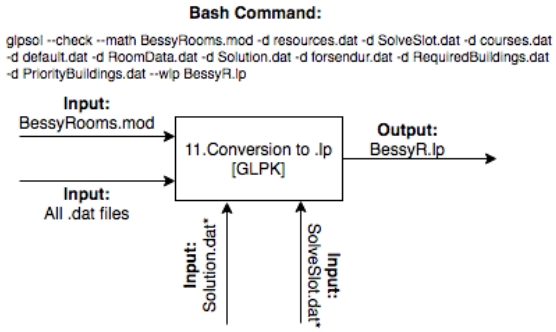

Figure B.18: Conversion of the data and the model to a lp file

12. Optimization
In this part, the optimization takes place using the solver Gurobi. As can be seen in Figure B.19 the input file is *BessyR.lp* which was constructed in part 11 and the output file is *BessyR.sol* which contains the optimal solution for the room scheduling as can be seen in Figure B.19 along with the bash command needed.

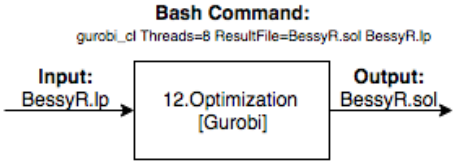

Figure B.19: The optimization for Phase II

13. Conversion of info
In part 12 the the optimization took place. When the optimization is finished, the results are written to the file *BessyR.sol*. The file includes results for all variables no matter if it equals to zero or other. The zero elements are not needed and therefore, that part of the data is filtered away in this step. The output is then written to the file $hfix.dat$. The flow can be seen in Figure B.20.

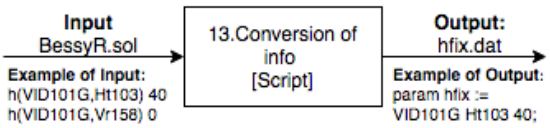

Figure B.20: The conversion of the lp file to a dat file which includes the solution

14. Print the solution for Phase II

In this part, the solution will be printed into the file *Solution.csv*. Here, all of the data (parts $1, 2..9$) and the model in part 10 along with the of the solution found in part 13 ($hfix.dat$). The GLPSOL solver will be used to solve the problem but since the optimal solution is imported, it will only take few seconds. Here, the statistics can be printed out in order to analyze the solution. The bash command needed to run this can be found in Figure B.21 along with the data.

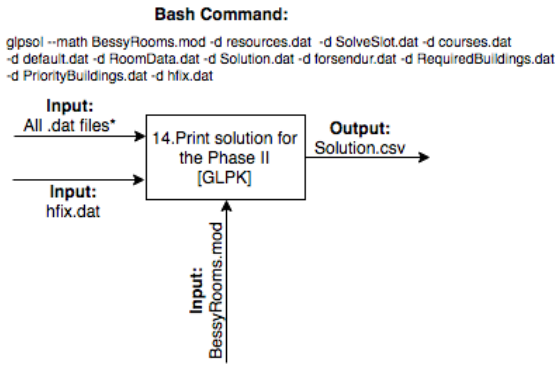

Figure B.21: The print phase for the room scheduling

15. Ugla

In this part the solution will be imported to UGLA examination timetabling visualizer by the director of examinations. The file *SolutionR.csv* will be read but it contains information about in which room/s a course is assigned to and how many students are assigned there. Here, the director of examinations reviews the solutions and can make changes if he would like to before publish. Since each slot is solved individually, all of the parts must be performed each time and imported one by one.

Figure B.22: Room scheduling import to UGLA