# Solving the open day timetabling problem using integer linear programming

BSc THESIS APPLIED MATHEMATICS

Dilan Gecmen

**Delft University of Technology**
**Faculty of Electrical Engineering, Mathematics and Computer Science**
**Delft Institute of Applied Mathematics**

# Solving the open day timetabling problem using integer linear programming

**(Dutch title : "De open dag rooster probleem oplossen met behulp van geheeltallig lineair programmeren" )**

A thesis submitted to the
Delft Institute of Applied Mathematics
in partial fulfillment of the requirements

for the degree

**BACHELOR OF SCIENCE**
**in**
**APPLIED MATHEMATICS**

**by**

**Dilan Gecmen**

**Delft, the Netherlands**
**August 2016**

**BSc THESIS APPLIED MATHEMATICS**

# “Solving the open day timetabling problem using integer linear programming”

**(Dutch title: “De open dag rooster probleem oplossen met behulp van geheeltallig lineair programmeren”)**

Dilan Gecmen

**Delft University of Technology**

August, 2016 Delft

# Preface

To many I am indebted for their support in various ways. However, above all there are a few to whom I am most obliged. First of all, I would like to thank my mentor for his incessant support. Without his generous guidance none of this would have been possible. Secondly, yet to no lesser extent, I want to thank my parents and sisters. Their unconditional and loving trust in my abilities have often helped me beyond the scope of their and my own knowledge.

*D.B. Gecmen*
*Delft, August 2016*

# Abstract

The timetabling problem is known as a large class of problems that fall under the mathematical field of scheduling. A widely used method to solve timetabling problems is the integer linear programming method, which we have used to solve the open day timetabling problem.

In this thesis we have addressed a timetabling problem for the open day at the Christian College Groevenbeek. This open day consists of two separate day parts: the morning and the afternoon. For both day parts we have received a data set. These data sets consist of students and their preferred studies.

To solve the problem we created several mathematical models formulated as an ILP. After that, we implemented these models in AIMMS to solve them for the data sets we have received. The first model we created was the feasibility model. We used this model to determine the appropriate number of lecture halls and lecture hall capacities when we have 4 rounds on both day parts. To achieve 4 rounds on both day parts we found that the best combination to have is 19 lecture halls with a capacity of 30 and 1 lecture hall with a capacity of 40.

In addition, for a good quality schedule objective functions were considered. The first objective was to minimize the number of presentations. The second objective was to minimize the total workload. The workload of a teacher is the total time a teacher is present at the college, which consists of the number of presentations he has to give and the number of gaps he has in his schedule. A gap is a round where a teacher is not scheduled to give a presentation, but has to be present. We added each objective and their corresponding constraints to the feasibility model. After applying both models to the data sets we concluded that the second objective resulted in a better schedule, as it achieves the theoretical minimum number of presentations and creates zero gaps in the schedules for both day parts.

When we combined the schedules for both day parts this did not result in a good schedule as some studies still contained gaps between the two day parts. To improve this we minimized the total workload for both data sets combined.

# CONTENTS

# 1 Introduction

The main goal of this thesis is to design schedules for open days where each student requires a tailored schedule taking into account the different requirements of such a schedule. Other timetabling problems such as school timetabling, employee timetabling, etcetera, share the same characteristics as the open days scheduling problem.

The timetabling problem is known as a large class of problems that fall under the mathematical field of scheduling. In recent years the timetabling problem has been greatly studied and published in the literature. The scheduling research community has developed many techniques that enable better solutions to practical scheduling problems, but there is still a lot that remains to be done.

In Section 1.1, we will give a detailed problem description about the open days timetabling problem considered in this thesis. In Section 1.2, an overview of existing literature about the timetabling problem will be given. From literature [1] we know that solving timetabling problems can be difficult. This is due to their complexity, which will be discussed in Section 1.3. A practical way to model the open day timetabling problem is as an integer linear program, see Section 1.4. In Section 1.5, we will explain several modeling techniques we use as we formulate our problem as an integer linear program. To implement the mathematical models and to visualize the data we make use of the software AIMMS, see Section 1.6. To actually solve the models AIMMS invokes the solver Cplex, see Section 1.4. Finally, in Section 1.7 an overview of the structure of this thesis will be given.

## 1.1. Problem description

Every year thousands of high school students visit the open days at universities to acquire knowledge about potential studies. As students attend an open day to acquire knowledge about potential studies, it is important to assure that each student attends their preferred studies at an open day. To achieve this each student needs to have a tailored schedule. When creating such a schedule one should take into account the number of available lecture halls, their capacity, the number of rounds, and many more. Because of this the open days scheduling problem is a complicated problem. For example, making a schedule for more than 1000 applicants is easier said than done.

Conventionally, making a schedule has been done manually, which takes a lot of manpower and time. We are going to use an optimization technique called integer linear programming, see Section 1.4, and the software AIMMS, see Section 1.6, which results in a good quality schedule.

The scheduling problem that we address in this thesis has arisen in the context of community colleges in the Netherlands. In the town Ermelo each year a general open day is organized at the Christian College Groevenbeek, which is a preparatory secondary vocational education. Teachers from several community colleges come here to give information sessions about studies provided by the community college they teach. There is a total of 32 different studies that are represented on this open day and there is a presentation about each study.

This open day, which consists of two separate day parts: the morning and the afternoon, is meant for students from 12 preparatory secondary vocational education schools in the region of Ermelo. Five of these schools are scheduled for the morning and seven are scheduled for the afternoon.

In the morning and afternoon the students will be picked up from their schools by buses, which brings them to the Christian College Groevenbeek, and at the end of the morning and afternoon they will be driven back. Students who attend the Christian College Groevenbeek will not be picked up as they already live nearby.

An open day gives students insight in potential studies, so it is important that each student is able to follow all their preferred studies. Presentations on studies, which take place at the same time, are all presentations about different studies. This is because there is just one teacher available for each study. Besides that, there are students who choose to follow more than one presentation. To assure that they can follow each presentation we need to have multiple rounds. A round consists of several different presentations, which take place at the same time in different lecture halls.

For the open day we do not want a lot of rounds on both day parts, additionally we do not want to use too many lecture halls, because that would defeat the purpose of being efficient.

The following scenario is set up for the purpose of determining the most efficient round division: each day part consists out of 4 hours, each round consists out of 50 minutes, and there are breaks in between each round to complete the 4 hour day part. So we want to make a tailored schedule that consists of 4 rounds for each day part.

To create a schedule consisting of 4 rounds we also need to find a suitable number of lecture halls with sufficient capacity taking into account the data we are given. First, to find the right number of lecture halls we are going to consider unlimited capacity in Section 3.2.1. There after, as lecture halls typically have limited capacity, we include these capacities in Section 3.2.1.

Once we have found a feasible solution we would like to improve the quality of this solution. In general, a schedule is definitely better when there are as little gaps as possible. As students will be stuck at the college for an entire day part, it does not matter if the schedule for each student contains gaps. On the other hand, the schedule for each teacher who gives presentations on a course needs to contain as little gaps as possible, because they do not want to spend their whole day at the school.

Because we have 4 rounds teachers who give presentations on studies with a lot of applicants will possibly give a presentation in each round in each day part. On the other hand, teachers with presentations on a study with not many applicants will give one, two or three presentations throughout the day. For these teachers we do not want these presentations to be at the beginning of the morning and at the end of the afternoon as this will result in a huge gap in their schedule. Instead, we want them to be at the end of the morning and at the beginning of the afternoon.

All these requirements and more will be fitted in mathematical models for the open days at the Christian College Groevenbeek in Chapter 2. In Chapter 3, these models will be used to determine the best schedule for the data we received.

## 1.2. LITERATURE OVERVIEW

The timetabling problem arises in many different settings, such as transportation, health care institutions, educational institutions, and sports. In 1996 Wren [2] defined timetabling as follows

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

Basic terms used in timetabling are given in Table 1.1.

Table 1.1: Basic terms used in timetabling

| **Terms** | **Definition** |
|---|---|
| Event | A scheduled activity. For example, examinations or courses. |
| Time slot | An interval of time. Each event can be scheduled in a time slot. |
| Resource | Events require resources. For example, lecture halls and equipment. |
| Constraint | A restriction to schedule events. For example, the capacity of a lecture hall is a restriction. |
| Individual | Someone who attends the events. |
| Conflict | Two events are scheduled in the same lecture hall in the same period. |

Constraints are restrictions to schedule events, which can be categorized in two groups: hard constraints and soft constraints. Hard constraints can never be violated. An example of a hard constraint is: two events can not be scheduled in the same room in the same period. Soft constraints are constraints we would like not to violate, because these constraints create a quality timetable. An example of a soft constraint is: the schedule needs to contain as little gaps as possible.

The timetabling problem contains a set of events, periods and hard constraints. The set of events are assigned to time slots, such that the hard constraints are not violated. The main goal of timetabling is to produce a feasible timetable.

In general, the timetabling problem refers to timetabling in educational institutions. In 1999 Schaerf [3] divided the timetabling problem in educational institutions into three groups: school timetabling, course timetabling, and examination timetabling.

The school timetabling problem consists of a set of courses, classes, weekly periods, and teachers. In this problem the main goal is to assign courses to periods and teachers to classes, while doing this the set of constraints can not be violated. An example of a constraint is lecture hall capacity.

In 1998 Carter and Laporte [4] defined course timetabling as follows

*"A multi-dimensional assignment problem in which students, teachers (or faculty members) are assigned to courses, course sections or lecture halls; events (individual meetings between students and teachers) are assigned to lecture halls and times."*

The main goal of course timetabling is to produce a feasible timetable, such that the constraints are not violated. These constraints can be divided in hard and soft constraints. Examples of hard constraints in course time tabling are: the number of students in a lecture hall can not exceed the capacity of the lecture hall, and a teacher can not be assigned to two courses in the same time slot. Example of a soft constraint is: teachers should not be scheduled to courses that take place in the first time slot and the last time slot as this results in a huge gap between courses.

Notice that the open day timetabling problem has a lot of similarities with the course timetabling problem. In both problems the goal is to make a schedule over a time period where students and teachers are assigned to lecture halls in a particular period in such way that mutual meetings can take place. Thus, the hard constraints of both problems are the same. The only difference can occur in the soft constraints.

When examinations need to be scheduled we have the examination timetabling problem. In 1998 Carter and Laporte [4] defined examination timetabling as follows

*"The assigning of examinations to a limited number of available time periods in such a way that there are no conflicts or clashes."*

The main goal of the examination timetabling is to produce a feasible timetable, such that the constraints are not violated. Just like the course timetabling the constraints can be divided in hard and soft constraints. Examples of hard constraints in examination timetabling are: lecture halls that are used for examinations need to have sufficient capacity, a student can not be scheduled for two examinations simultaneously, and many more. An example of a soft constraint is: students should not have more than two examinations at the same day.

School timetabling, course timetabling, and examination timetabling, share the same basic characteristics of the general timetabling problem. Besides that, each one of them is different, because each problem has its own requirements and rules.

In recent years, a lot of methods are published to model and solve timetabling problems. Based on this literature we can conclude that modeling and solving the timetabling problems can be difficult. Modeling is difficult, because one needs to find a way to represent all the requirements as a mathematical model. Besides that, solving a model can be difficult, because of its complexity, see Section 1.3.

One way to model a timetabling problem is using graphs, which was shown in 1985 by de Werra [5]. In 2006 S. Abdullah [6] presented a graph coloring model that can be used to model and solve the school timetabling problem, see Figure 1.1. In graph theory, the graph coloring problem is to assign colors to vertices of a graph such that nodes that share an edge do not have the same color. The minimum number of colors to do this is called the chromatic number. Graph coloring is an old and well known problem. It all started in the 1800s with the problem of coloring each country on a map in such way that two countries that have the same border do not receive the same color. If we represent each country as a vertice and each edge as a border between two countries, we obtain a planar graph and we may solve the problem as a graph coloring problem.

Let us give an example of solving a particular school timetabling problem using the graph coloring method. The graph that is illustrated in Figure 1.1 contains five nodes, which represent the five different courses: A, B, C, D, and E. Suppose we want to find the minimal number of time slots that we need to schedule all the courses. If there is an edge between two nodes, this means that these two courses can not be scheduled at the same time. From 1.1 we can see that course A can not be scheduled at the same time as courses B and C, course B can not be scheduled at the same time as courses A and D, course C can not be scheduled at the same time as courses A and E, course D can not be scheduled in at the same time as courses B and E, and finally course E can not be scheduled at the same time as courses C and D. To divide the courses into time slots we color the vertices of the graph in such a way that if there is an edge between two nodes these two nodes have different colors, see Figure 1.1. The chromatic number is 3. Thus, the minimal number of time slots we need to schedule in the courses is 3.

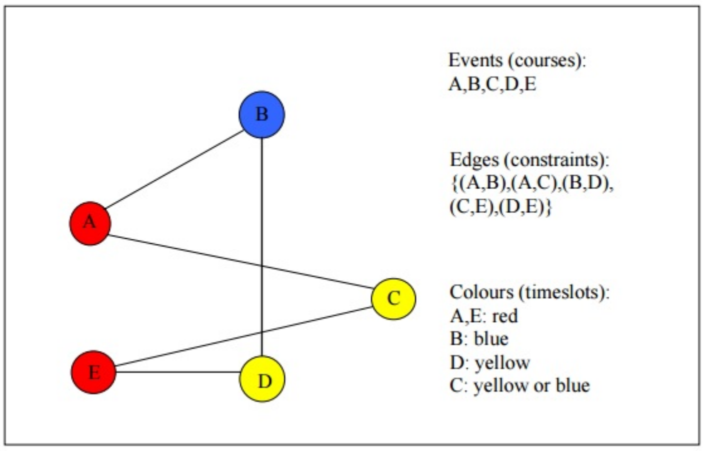

Figure 1.1: A graph model for a simple course timetabling problem

In this thesis, we are not going to use the graphical setting. As the open day timetabling problem involves large data sets and this is not efficient to model as a graph. Another more practical way to model the open day timetabling problem is as an integer linear program, see Section 1.4.

## 1.3. COMPLEXITY THEORY

Complexity theory is the theory of classifying optimization problems on how difficult they are to solve, where a distinction is made between optimization problems and decision problems they are related to. An optimization problem is finding the best solution in a set of feasible solutions and a decision problem is a problem that can be answered by either ’yes’ or ’no’.

For example the traveling salesman problem (TSP), the optimization problem searches a tour in a graph of shortest length for the salesman between $n$ cities. In the decision problem, the question is whether there exists a tour in a graph through $n$ cities with length at most $l$.

To solve a traveling salesman problem different algorithms can be applied to find a solution. In general, an algorithm is a finite set of instructions that defines a sequence of operations to solve a particular computational problem for each instance of a given problem class. An algorithm must possess the following properties:

- *Finiteness:* After a finite number of steps an algorithm must terminate.
- *Definiteness:* Each step of an algorithm must be defined precisely.
- *Inputs:* An algorithm has zero or more inputs, which are taken from specified sets of objects. These inputs are values and are supplied either before the algorithm starts or as the algorithm runs.
- *Outputs:* An algorithm has one or more outputs. These outputs are values which are determined by the inputs.
- *Effectiveness:* In practice, each step of the technique has to be feasible.

Besides the previous described properties, an algorithm should of course also be correct, which means that it should solve a problem correctly for each instance. Moreover, an algorithm needs to be efficient.

It is important to know that the running time of an algorithm depends on the size of the input. We can think of the running time as the length of time taken to run the program on some standard computer. If the running time of an algorithm is some polynomial function of the size of the input, for example if it runs in linear, quadratic or cubic time, then the algorithm runs in polynomial time and the problem it solves is in class P. The algorithm gives the solution in at most $a \times n^c$ steps where $a \geq 0$, $c \geq 0$, and $n$ is the length of the input.

A problem is in class NP if we can verify a ’yes’ answer with a polynomial time algorithm. This does not mean it is solvable in polynomial time. If you can not verify a solution of a problem you certainly can not solve the problem. This implicates that the complexity class P is contained in the complexity class NP, see Figure 1.2. For example, the traveling salesman problem is in NP. This means if there is a tour in a graph of length at most $l$ we can verify this in polynomial time.

The class NP-complete contains the hardest problems of the class NP. To be exact, all problems in NP can be reduced to a problem that is NP-complete. If there exists a polynomial algorithm for a NP-complete problem, this implicates that P=NP, see Figure 1.2 [7]. It is not known if P=NP, but most scientists believe that P≠NP, which means that there are no efficient algorithms to solve NP-complete problems.

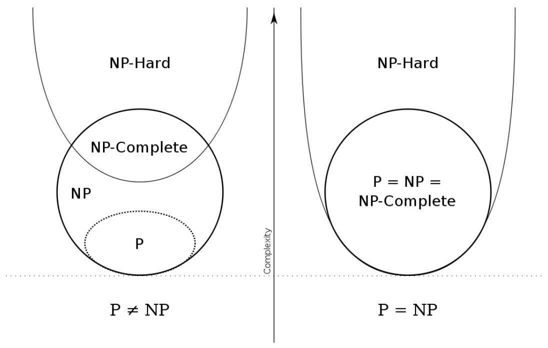

Figure 1.2: P versus NP.

## 1.4. LINEAR PROGRAMMING

A widely used method to solve timetabling problems is linear programming, which we will use to solve the open day timetabling problem in this thesis.

The mathematical model for the timetabling problem for the open day consists of requirements that are represented by linear relationships. A method to achieve the best outcome in such a model is linear programming (LP). The objective of a linear programming problem is a linear function that can be maximized or minimized. The problem is usually expressed in canonical form as

$$\begin{aligned} \min \quad & \mathbf{c}^{\mathrm{T}}\mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} \leq \mathbf{b}, \\ & \mathbf{x} \geq \mathbf{0}, \\ & \mathbf{x} \in \mathbb{R}^{+} \end{aligned}$$

Here, the input consists of an $n$-dimensional vector $\mathbf{c}$, an $m$-dimensional vector $\mathbf{b}$, and an $m \times n$-matrix $\mathbf{A}$. The vector $\mathbf{x}$ represents the vector of variables that has to be determined. Note that s.t. stands for subject to and min stands for minimize.

The linear programming problem can be solved in polynomial time, i.e. of complexity class P. Thus a linear program can be solved efficiently, for example by using the ellipsoid method [8] or the interior point method [9]. The simplex method [10], which is remarkably efficient in practice, is often used for solving linear programming problems. But it is still not known if it is polynomial in the worst-case scenario.

In dimension two and possibly in dimension three you can also solve linear programming graphically. The graphical method gives insight about what a linear program exactly is, but does not work in higher dimensions.

For example, suppose we want to find the optimal solution of the following LP

$$\begin{aligned}
\max \quad & \mathbf{x}_1 + \mathbf{x}_2 \\
\text{s.t.} \quad & \mathbf{x}_1 + 2\mathbf{x}_2 \leq 4 \\
& 4\mathbf{x}_1 + 2\mathbf{x}_2 \leq 12 \\
& -\mathbf{x}_1 + \mathbf{x}_2 \leq 1 \\
& \mathbf{x}_1 \geq 0 \\
& \mathbf{x}_2 \geq 0
\end{aligned}$$

In Figure 1.3 the dashed line indicates the objective function and the set of feasible solutions is depicted in yellow. In general, if the constraint set is bounded the objective function takes its maximum (or minimum) in one of the corner points. See Figure 1.3 for the optimal point.

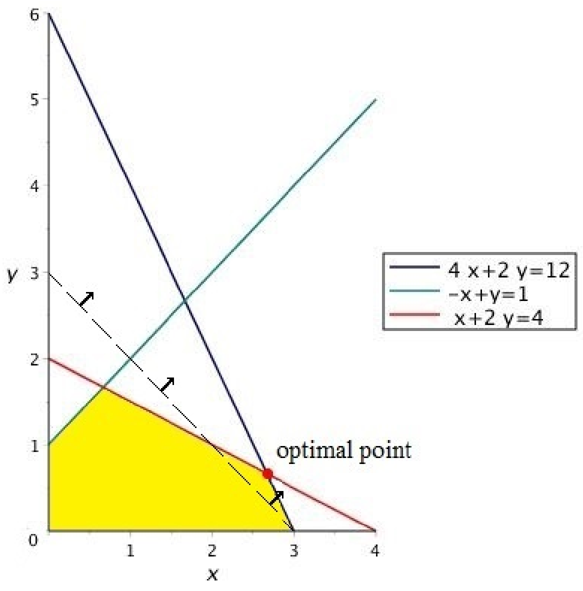

Figure 1.3: Graphical method of finding the optimal solution

In many practical applications of linear programming the variables are restricted to integer values. A linear problem with the additional constraint that all variables are integers is called an integer linear program (ILP). The problem is usually expressed in canonical form as

$$\begin{aligned}
\min \quad & \mathbf{c}^{\mathrm{T}}\mathbf{x} \\
\text{s.t.} \quad & \mathbf{A}\mathbf{x} \leq \mathbf{b}, \\
& \mathbf{x} \geq \mathbf{0}, \\
& \mathbf{x} \in \mathbb{Z}^n
\end{aligned}$$

Here, the input consists of an $n$-dimensional vector $\mathbf{c}$, an $m$-dimensional vector $\mathbf{b}$, and an $m \times n$-matrix $\mathbf{A}$. The vector $\mathbf{x}$ represents the vector of variables that has to be determined.

In general, the ILP problem is NP-hard, thus any known solution method to solve ILP problems is non-polynomial. Methods that are commonly used include the Branch-and-Bound method and the cutting-plane method [11].

When a LP is derived from an ILP by removing the constraint that all variables have to take integer values the result is called the LP relaxation of the original problem. A lot of solution methods to solve ILP problems make use of the LP relaxation of the problem. Note that in the LP we minimize the same objective function, but over a larger set of solutions, thus $z_{LP} \leq z_{ILP}$, where $z_{LP}$ is the optimal solution of the LP relaxation and $z_{ILP}$ is the optimal solution of the ILP. We can see that each optimal solution of the LP relaxation provides a lower bound on the optimal solution of the ILP.

The gap between the optimal LP relaxation value and the optimal ILP solution is called the integrality gap (IG) of the linear program. For minimization problems the integrality gap is defined as $\sup_I \frac{z_{ILP}(I)}{z_{LP}(I)}$, where $I$ stands for a particular instance of the problem. In a maximization problem the fraction is reversed. The integrality gap takes the following values:

$$IG = \begin{cases} \geq 1 & \text{if} \quad z_{LP} \leq z_{ILP} \\ = 1 & \text{if} \quad z_{LP} = z_{ILP} \end{cases}$$

The Branch-and-Bound algorithm depends highly on the fact that the LP relaxation is solvable. First, the LP relaxation is solved. After that, if one of the decision variables is fractional, the algorithm divides the feasible region in two subdivisions/subproblems. Each of these subproblems have an additional constraint on the value of the fractional variable, this does not exclude any other integer solutions. The previous approach of dividing the problem into subproblems is applied to the two subproblems.

An important aspect of the Branch-and-Bound algorithm is pruning, which is cutting off and discard nodes when you can prove that the node, or any of its descendants, will never be optimal or feasible. The method has three ways of pruning subproblems. Firstly, we have prune by infeasibility. This means that the LP relaxation has no feasible solution, as a consequence the ILP does not have a feasible solution. Secondly, we prune if the optimal solution of the LP relaxation has an integer value. The optimal solution for the LP relaxation of the subproblem provides an upperbound on the value of the optimal solution of the original problem. Finally, we prune if the optimal solution of the LP relaxation guarantees that there does not exists a better solution in the subproblem for the original problem. This is done by recording bounds on the optimal solution. This way of pruning can influence the running time of the algorithm, thus it is important to have a strong LP relaxation. We have a strong LP relaxation when the integrality gap is close to one, which can be achieved by adding valid inequalities to the ILP.

One way to find valid inequalities is by using Gomory's cutting plane method is. Gomory cuts are used to reduce the feasible region of the problem. Adding these cuts to the problem gives us a stronger LP relaxation.

For integer linear programming problems optimization solvers exist that are commonly used. The Gurobi optimizer [12] and Cplex optimizer [13] are examples of commonly used optimization solvers for linear programming (LP), integer linear programming (ILP), and mixed integer linear programming (MILP). Integer linear programming problems are generally solved by Cplex and Gurobi by using linear-programming based on a combination of the Branch-and-Bound algorithm and cutting planes as this results in a stronger formulation and a smaller integrality gap.

In this thesis we want our variables to be a special kind of integer variables, namely binary variables, which take the value 0 or 1. Because the variables we want to use are binary variables the method that is used to solve the problem is integer linear programming (ILP). This is because the sets of classrooms, rounds, students, and study information sessions are assigned to each other.

The problem of event planning is in some sense comparable to the assignment problem, which is a special case of an integer linear programming problem. The Hungarian method [14] is a combinatorial optimization algorithm that solves the assignment problem in polynomial time.

A common example to explain the assignment problem is the following: Let there be $n$ workers and $m$ jobs with $n = m$. Any worker can be assigned to do any job and each worker-job pair has a certain cost. Further, each worker must be assigned to exactly one job and each job must be assigned to exactly one worker. This is done in a way such that the total cost is minimized or maximized.

Mathematically an assignment is a one-to-one mapping between the elements of two finite sets of equal sizes. There are different ways to formally represent an assignment. One way is by a permutation. A permutation $\phi$ is a bijective mapping of a finite set $N = \{1, 2, \ldots, n\}$ into itself which corresponds in a unique way with an adjacency matrix $X_\phi = (x_{ij})$ with

$$x_{ij} = \begin{cases} 1 & \text{, if items match} \\ 0 & \text{, if items don't match} \end{cases}$$

The corresponding graph is a bipartite graph $G = (V, W; E)$ where $V$ and $W$ are sets of vertices who have the same number of vertices, thus $|V| = |W| = n$. An edge $(i, j) \in E$ occurs if and only if the items match. See Figure 1.4 for an example of a permutation, the corresponding adjacency matrix and bipartite graph.

$$\phi = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 3 & 2 & 4 & 1 \end{pmatrix}$$

$$X_\phi = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

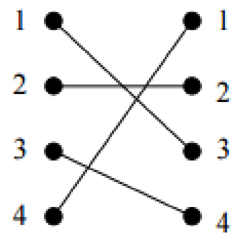

Figure 1.4: Representations of assignments

As previously stated, the assignment problem is a special kind of integer linear programming problem. Thus, the assignment problem can be expressed as a integer linear programming problem in the following form

$$\begin{aligned} \min \quad & \sum_{i \in A} \sum_{j \in T} C(i, j) x_{ij} \\ \text{s.t.} \quad & \sum_{j \in T} x_{ij} = 1 \text{ for } i \in A \\ & \sum_{i \in A} x_{ij} = 1 \text{ for } j \in T \\ & x_{ij} \in \{0, 1\} \text{ for } i, j \in A, T \end{aligned}$$

The variable $x_{ij}$ represents the assignment of worker $i$ to job $j$. If the assignment is done $x_{ij}$ takes the value 1 and if the assignment is not done the value of $x_{ij}$ is 0. The first constraint requires that each worker is assigned to exactly one job, and the second constraint requires that each job is assigned to exactly one worker.

## 1.5. INTEGER LINEAR PROGRAMMING TRICKS

To formulate our problem as a mathematical model we use several modeling techniques in Chapter 2. In this section some of these techniques to create constraints that satisfy the desired properties are discussed.

For example, let us say we have a set of students given by $I = \{1 \ldots n\}$ and a set of free electives given by $J = \{1 \ldots m\}$. Each student $i \in I$ needs to choose one free elective $j \in J$ to follow besides their regular study program. We denote this input by

$$a_{ij} = \begin{cases} 1 & \text{if student } i \text{ chooses free elective } j \\ 0 & \text{otherwise} \end{cases}$$

We want to know for each free elective if it needs to be teached or not. This is represented by the decision variable $y_j$ as

$$y_j = \begin{cases} 1 & \text{if free elective } j \text{ is given} \\ 0 & \text{otherwise} \end{cases}$$

In order to assure the right value for $y_j$, we check if the input $a_{ij}$ is 0 or 1. If the input is zero for a specific free elective for each student, then this free elective will not be given. If the input $a_{ij}$ is 1 for some $i \in I$ and $j \in J$, then the free elective is given and $y_j = 1$. To create the desired properties the following constraint can be used

$$a_{ij} \leq y_j$$

In Chapter 2 we will use a similar technique as the previous example to give the right value to a particular binary variable.

To find an upper bound or lower bound of a variable we use Big M constraints. For example, let us say we have the following cost function

$$c(x) = \begin{cases} k + cx & \text{if } x > 0 \\ 0 & \text{if } x = 0 \end{cases}$$

Here, $k$ stands for the start-up cost and $c$ stands for the per-unit cost. Let us say we have a variable $y$ that indicates whether $x > 0$. Now we have the objective function

$$c(x) = ky + cx$$

and we add the following constraint

$$x \geq My$$

Here, $M$ is a sufficiently large constant, this means that $M$ is larger then or equal to the largest value that the variable $x$ can take in a feasible solution. Note that we need to find an appropriate value for $M$, as a too small or too large value results in a weak LP relaxation.

To find an optimal solution an integer linear programming problem enumeratively searches the feasible region. The direction of the search is terminated when an optimal integer solution is found, or when the objective function is found to go below a specific value, or when there is no feasible integer solution. This process of searching an optimal integer solution is speed up when as many constraints as possible are imposed, which define the feasible and the optimal region. Each of these constraints help to define and reduce the feasible solution region. These constraints can appear unnecessary, but they can decrease the solution time of an integer linear programming problem. As we have seen that Branch-and-Bound performs better with tighter LP relaxations it can be useful to replace a constraint by multiple constraints. For example, in 1978 H.P. Williams [15] showed that by replacing constraints with an indicator variable like

$$x_1 + x_2 - Md \geq 0$$

by more constraints

$$\begin{aligned} x_1 - Md &\geq 0 \\ x_2 - Md &\geq 0 \end{aligned}$$

the running time will be speed up.

## 1.6. AIMMS

To implement the mathematical models we formulated in Chapter 2 and to visualize the data we make use of the software AIMMS. AIMMS ("Advanced Interactive Multidimensional Modeling System") is a software system for modeling and solving large-scale optimization and scheduling-type problems. It is meant to implement a problem easily and to visualize the data.
Optimization problems that AIMMS provides support to are

- Complementarity problems
- Constraint programming
- Global optimization
- Linear programming
- Mixed-integer programming
- Mixed-integer nonlinear programming
- Nonlinear programming
- Quadratic programming
- Robust optimization
- Stochastic programming

To solve a problem AIMMS invokes different solvers for different types of problems. For example, for integer linear programming problem the standard solvers that AIMMS invokes are Gurobi and Cplex, and for nonlinear programming AIMMS invokes the solver SNOPT.

In this thesis we will solve the timetabling problem using the solver Cplex. When solving an ILP, we can supply hints to help CPLEX find a solution. These hints consist of value assignments for a set of variables, known as an advanced start, or a warm start. A warm start can improve the CPLEX solve process dramatically. In particular, a warm start can be used to help guide the solver towards a first solution, or to start search from a known solution to further improve it. In Section 3.2.2 we will provide the engine with an initial solution of our timetabling problem to improve the provided solution.

## 1.7. THESIS STRUCTURE

First, in Chapter 2, the formulation of our problem as an integer linear programming problem will be given. To give a mathematical model of our problem as an ILP we need to set up constraints that satisfy each requirement we have. Doing so will create a feasibility model. After we determined this we want to create a good quality schedule. For this we add two objective functions with associated constraints to the feasibility model. The first one minimizes the number of presentations and the second one minimizes the total workload.

Thereafter, in Chapter 3 a detailed description of the data used for creating a tailored schedule will be given. The feasibility model we formulated in Chapter 2 will be used to consider different scenarios. We use this model to determine the number of lecture halls and their capacities when we have day parts consisting of 4 rounds. To eventually present the best possible schedule we apply the models with an objective function to the data sets we have received. To help illustrate the results, tables are available and displayed.

Finally, conclusions are drawn and stated in Chapter 4 and a summary of all results is given to get a good overview of the results of this bachelor project.

# 2
# INTEGER LINEAR PROGRAMMING FORMULATION

Integer linear programming is the method we will use to solve the open days scheduling problem, as we already explained in Section 1.4. First, in Section 2.1 we will create a feasibility model with no objective function. Also each constraint of this model will be explained and a mathematical formulation will be given. Eventually, we are going to use the feasibility model in Chapter 3 to find an optimal solution for the data given a number of rounds, lecture halls, and lecture hall capacities.

As we will use the feasibility model to compute the minimum number of required rounds for a given number of lecture halls it is easier to temporarily add an objective function and a constraint, see Section 2.2. Now we can calculate the minimum number of rounds in one run instead of checking the feasibility given a fixed number of rounds.

As we already mentioned, the main goal of this bachelor project is to create a good quality schedule. But what makes a good schedule? For example, a schedule with a lot of gaps is not considered as a good schedule, because this will result in a lot of breaks between rounds for teachers. So we want to create a schedule with as little gaps as possible. To do this we are going to minimize the total number of presentations on each study, see Section 2.3.1. Besides that, we are going to minimize the workload, which is the number of rounds a teacher has to be present, see Section 2.3.2. Note that if a teacher needs to wait one or more rounds between two rounds, where he is scheduled to give a presentation, then these rounds also count as present. We call rounds where a teacher is not scheduled, but needs to be present, gaps.

## 2.1. FEASIBILITY MODEL

The scheduling problem that we address in this thesis has arisen in the context of community colleges in the Netherlands. In the town Ermelo every year a general open day is organized at the Christian College Groevenbeek, which is a preparatory secondary vocational education.

Students can choose to follow presentations on several different studies offered at community colleges in the Netherlands. The students are given by the set $I = \{1 \ldots n\}$ and the set of studies they can choose from is given by $J = \{1 \ldots m\}$. Note that the set of studies is nothing else then the set of teachers, as each teacher represents just one study.

To assure that each student is able to follow each preference we need a number of rounds, which is given by the set $K = \{1 \ldots r\}$. A round consists of several different presentations and each presentation is about a different study. We also need the right number of lecture halls, which is given by the set $L = \{1 \ldots p\}$.

For each student $i \in I$ we need to know which study $j \in J$ they have chosen. We denote this input by

$$a_{ij} = \begin{cases} 1 & \text{if student } i \text{ has chosen study } j \\ 0 & \text{otherwise} \end{cases}$$

We want to assign each student $i \in I$ and each of their preferred studies $j \in J$ to a round $k \in K$. This assignment is represented by the decision variable $x_{ijk}$ as

$$x_{ijk} = \begin{cases} 1 & \text{if student } i \text{ follows study } j \text{ in round } k \\ 0 & \text{otherwise} \end{cases}$$

We also want to assign each study to a lecture hall and round. This assignment is represented by $y_{jlk}$, which is also a decision variable, as

$$y_{jlk} = \begin{cases} 1 & \text{if study } j \text{ is given in lecture hall } l \text{ in round } k \\ 0 & \text{otherwise} \end{cases}$$

When we assign a study to a lecture hall we need to assure that the capacity of the lecture hall is sufficient. For this we introduce $c_l$ with $l \in L$, which denotes the capacity of lecture hall $l$.

The open day gives students insight about potential studies. Therefore we want a schedule where each student is able to follow each of their preferred studies. In the formulation we do this by checking each round on whether a student is assigned to a particular study. This is done for each student and for each of his or her preferences. Now we can formulate the requirement as follows

$$\sum_{k=1}^{r} x_{ijk} = a_{ij} \qquad \forall i \in I, \quad \forall j \in J \tag{2.1}$$

For each study there is just one teacher available. Thus, in each round there can be at most one presentation for each study. We can obtain this for each study by checking the sum over all lecture halls in each round to make sure that the sum is at most 1. Now we can formulate the requirement as follows

$$\sum_{l=1}^{p} y_{jlk} \leq 1 \qquad \forall k \in K, \quad \forall j \in J \tag{2.2}$$

Now we obtained that each study can only be given once in a round. But, we also want to assure that a study is given in at most one lecture hall in each round. To do this we check for each round and lecture hall if the sum over all the studies is at most 1. This gives us the following constraint

$$\sum_{j=1}^{m} y_{jlk} \leq 1 \qquad \forall l \in L, \quad \forall k \in K \tag{2.3}$$

We do not want a student to be scheduled to follow two different presentations in one round, because as we already said we want them to be able to acquire knowledge about each study they have chosen. To assure that each student follows just one presentation in a round we check in each round and for each student if the sum of all studies is at most 1. The requirement is given by the following constraint

$$\sum_{j=1}^{m} x_{ijk} \leq 1 \qquad \forall k \in K, \quad \forall i \in I \tag{2.4}$$

Also, a student cannot be assigned to a presentation in a round if that particular presentation is not given in that round. To check this we have the following constraint

$$x_{ijk} \leq \sum_{l=1}^{p} y_{jlk} \qquad \forall i \in I, \quad \forall j \in J, \quad \forall k \in K \tag{2.5}$$

Notice that this constraint enforces the link between the decision variables $x$ and $y$.

As each lecture hall has a certain capacity the number of students in each lecture hall cannot exceed the lecture halls capacity during each round. To determine the number of students scheduled for a certain study in a round we check the total number of the students for each study in each round. After that we check if this number is not higher than the capacity of the lecture hall the study is assigned to. This requirement is given by the following constraint

$$\sum_{i=1}^{n} x_{ijk} \leq \sum_{l=1}^{p} y_{jkl} \times c_l \qquad \forall k \in K, \quad \forall j \in J \tag{2.6}$$

Notice that constraint (2.6) also makes sure that a student cannot be assigned to a presentation in a round if that particular presentation is not given that round. This means that constraint (2.5) is not necessary, but we will still use constraint (2.5).

As already explained, solving the program will be done by CPLEX which uses a combination of the Branch-and-Bound method and the cutting-plane method. Adding as many, and not redundant, constraints as possible results in a stronger LP relaxation. Thus, the process of finding an optimal solution will be sped up. This is the reason why we make use of constraint (2.5).

We can now represent all of the previous in the following feasibility model, which is also an ILP

$$
\begin{aligned}
\min \quad & 0 & & \qquad (1.1)\\
\text{s.t.} \quad & \sum_{k=1}^{r} x_{ijk} = a_{ij} & & \forall i \in I, \quad \forall j \in J\\
& \sum_{l=1}^{p} y_{jlk} \leq 1 & & \forall j \in J, \quad \forall k \in K\\
& \sum_{j=1}^{m} y_{jlk} \leq 1 & & \forall l \in L, \quad \forall k \in K\\
& \sum_{j=1}^{m} x_{ijk} \leq 1 & & \forall i \in I, \quad \forall k \in K\\
& x_{ijk} \leq \sum_{l=1}^{p} y_{jlk} & & \forall i \in I, \quad \forall j \in J, \quad \forall k \in K\\
& \sum_{i=1}^{n} x_{ijk} \leq \sum_{l=1}^{p} y_{jkl} \times c_l & & \forall j \in J, \quad \forall k \in K\\
& x_{ijk} \in \{0,1\} & & \forall i \in I, \quad \forall j \in J, \quad \forall k \in K\\
& y_{jlk} \in \{0,1\} & & \forall j \in J, \quad \forall l \in L, \quad \forall k \in K
\end{aligned}
$$

## 2.2. MINIMIZE TOTAL NUMBER OF ROUNDS

In Chapter 3, we will use the feasibility model (1.1) to calculate the required number of rounds, lecture halls and capacity for the data sets we received. Calculating the minimal number of rounds for a different number of lecture halls can be done by iteratively solving model (1.1). We can also calculate the minimum number of rounds for a given number of lecture halls in one run. So for computational convenience we temporarily add a new variable, an objective function, and a constraint. The variable we introduce to do this is

$$p_k = \begin{cases} 1 & \text{if round } k \text{ is used} \\ 0 & \text{otherwise} \end{cases}$$

In order to assure the right value for $p_k$, we check if the variable $y_{jlk}$ is 0 or 1. If this variable is zero for all studies in a specific round, none of the studies is assigned to be scheduled in that round, thus the round is not used. If $y_{jlk}$ is 1 for some $j \in J$, $l \in L$ then the round is used and $p_k = 1$. This requirement is given by the following constraint

$$y_{jlk} \leq p_k \qquad \forall j \in J, \quad \forall l \in L, \quad \forall k \in K$$

The objective is to minimize the total number of rounds and is given by

$$\min \sum_{k=1}^{r} p_k$$

Adding the variable, objective, and constraint to model (1.1) we get the following ILP

$$
\begin{aligned}
\min \quad & \sum_{k=1}^{r} p_k & & \qquad (1.2)\\
\text{s.t.} \quad & y_{jlk} \le p_k & & \forall j \in J, \quad \forall l \in L, \quad \forall k \in K\\
& \sum_{k=1}^{r} x_{ijk} = a_{ij} & & \forall i \in I, \quad \forall j \in J\\
& \sum_{l=1}^{p} y_{jlk} \le 1 & & \forall j \in J, \quad \forall k \in K\\
& \sum_{j=1}^{m} y_{jlk} \le 1 & & \forall l \in L, \quad \forall k \in K\\
& \sum_{j=1}^{m} x_{ijk} \le 1 & & \forall i \in I, \quad \forall k \in K\\
& x_{ijk} \le \sum_{l=1}^{p} y_{jlk} & & \forall i \in I, \quad \forall j \in J, \quad \forall k \in K\\
& \sum_{i=1}^{n} x_{ijk} \le \sum_{l=1}^{p} y_{jkl} \times c_l & & \forall j \in J, \quad \forall k \in K\\
& x_{ijk} \in \{0,1\} & & \forall i \in I, \quad \forall j \in J, \quad \forall k \in K\\
& y_{jlk} \in \{0,1\} & & \forall j \in J, \quad \forall l \in L, \quad \forall k \in K\\
& p_k \in \{0,1\} & & \forall k \in K
\end{aligned}
$$

## 2.3. Objective functions

In sections 2.1 and 2.2 we created feasibility models which can determine the required number of rounds, lecture halls, and lecture hall capacities. Using this information we eventually want to create a good schedule. We can achieve this by adding objective functions and constraints to model (1.1).

One way to make a good schedule is to minimize the total number of presentations. If we minimize the number of presentations of a particular study the total workload of a teacher would also be minimized, which is convenient for the teacher. Besides that, if a study with not many applicants is scheduled for 3 rounds in each day part the lecture halls in each round would not be as crowded as we wish. By minimizing the number of presentations we can have lecture halls that are better utilized.

Another way to make a good schedule is to minimize the number of gaps between two rounds where a teacher is scheduled to give presentations. A gap is a round where a teacher is not scheduled to give a presentation. Thus we want to minimize the total time length a teacher is scheduled. This is convenient for a teacher as he does not want to spent his entire day at the college if he is only scheduled to give two presentations.

In Sections 2.3.1 and 2.3.2 we will create mathematical models of the two ways we presented to create a better schedule. These models are formulated as integer linear programs.

### 2.3.1. Objective 1: Minimize number of presentations for each study

As we already explained, the first objective is to minimize the total number of presentations of each study. This is simply done by minimizing the total sum of the presentations given. We can do this by minimizing the total sum of the variable

$$
y_{jlk} = \begin{cases} 1 & \text{if study } j \text{ is given in classroom } l \text{ in round } k \\ 0 & \text{otherwise} \end{cases}
$$

The objective function is

$$
\min \quad \sum_{j=1}^{m} \sum_{l=1}^{p} \sum_{k=1}^{r} y_{jlk}
$$

Adding the objective function to model (1.1) gives the following ILP

$$
\begin{aligned}
\min \quad & \sum_{j=1}^{m}\sum_{l=1}^{p}\sum_{k=1}^{r} y_{jlk} && && (1.3)\\
\text{s.t.} \quad & \sum_{k=1}^{r} x_{ijk} = a_{ij} && \forall i \in I, \quad \forall j \in J\\
& \sum_{l=1}^{p} y_{jlk} \leq 1 && \forall j \in J, \quad \forall k \in K\\
& \sum_{j=1}^{m} y_{jlk} \leq 1 && \forall l \in L, \quad \forall k \in K\\
& \sum_{j=1}^{m} x_{ijk} \leq 1 && \forall i \in I, \quad \forall k \in K\\
& x_{ijk} \leq \sum_{l=1}^{p} y_{jlk} && \forall i \in I, \quad \forall j \in J, \quad \forall k \in K\\
& \sum_{i=1}^{n} x_{ijk} \leq \sum_{l=1}^{p} y_{jkl} \times c_l && \forall j \in J, \quad \forall k \in K\\
& x_{ijk} \in \{0,1\} && \forall i \in I, \quad \forall j \in J, \quad \forall k \in K\\
& y_{jlk} \in \{0,1\} && \forall j \in J, \quad \forall l \in L, \quad \forall k \in K
\end{aligned}
$$

### 2.3.2. OBJECTIVE 2: MINIMIZE WORKLOAD FOR EACH TEACHER

In general, a schedule is definitely better when there are as little gaps as possible. As for the students, it does not matter if the schedule for each student contains gaps since they will be stuck at the college for an entire day part, because they need to wait for the buses. On the other hand, the schedule for each teacher who gives presentations on a study needs to contain as little gaps as possible, because they do not want to spend their whole day at the school.

Thus, when a teacher is scheduled to give more than one presentation, we want him to be scheduled in rounds that are one after another. In other words, we want to minimize the workload for each teacher. The workload of a teacher is the total time a teacher is present at the college, which consists of the number of presentations he has to give and the number of gaps he has in his schedule. To give a precise definition, a gap is a round where a teacher is not scheduled to give a presentation, but has to be present.

Note that each teacher gives presentations about just one specific study. To minimize the total workload we need to determine the first and last round in which a teacher is scheduled. First, we want to know for each study if it is given or not in a round. The variable we introduce to determine this is

$$
t_{jk} = \begin{cases} 1 & \text{if study } j \text{ is given in round } k \\ 0 & \text{otherwise} \end{cases}
$$

In order to assure the right value for $t_{jk}$, we check if the variable $y_{jlk}$ is 0 or 1. If this variable is zero for a particular study in a specific round, then the study is not assigned to be scheduled in that round. If $y_{jlk}$ is 1 for some $j \in J$ and $k \in K$, then the study is assigned to be scheduled in that round. To create the desired properties the following constraint is used

$$
y_{jlk} \leq t_{jk}
$$

To determine the first round each study is given we introduce the variable $o_j$, which is a positive integer valued variable. The variable $o_j$ stands for the lower bound of each study $j$, thus the first round a presentation is given. To determine the last round each study is given we create variable $b_j$, which is also a positive integer valued variable. The variable $b_j$ stands for the upper bound of each study $j$, thus the last round a presentation is given.

To determine the lower bound we check for each round $k$ if the study is given , we set $o_j \leq k$. If the study is not given in a round, we set $o_j \leq M$. Note that we make use of Big $M$ constraints, as introduced in Section 1.5. We choose $M = 4$, which is sufficiently large, because the maximum number of rounds is 4. To create the desired properties the following constraint is used to determine the lower bound

$$
o_j \leq k + 4(1 - t_{jk})
$$

To determine the upper bound we check for each round $k$ if the study is given or not. If the study is given, then we set $b_j \geq k$, and if the study is not given we set $b_j \geq 0$. These requirements are given in the following constraint

$$b_j \geq k \times t_{jk}$$

Eventually, we want to minimize the difference between the upper bound and the lower bound. To achieve this we subtract the lower bound from the upper bound for each study and minimize the total sum. This objective function is given as

$$\min \quad \sum_{j=1}^{m} b_j - oj$$

Note that we are actually interested in minimizing $b_j - o_j + 1$, but adding 1 does not change the optimal decision and can therefore be omitted. Since the minimization will be done for each study, 32 times, "+32" will be added to the optimal solution the solver will find to get the optimal workload.

Adding the objective function and constraints to model (1.1) gives the following ILP

$$\begin{aligned}
\min \quad & \sum_{j=1}^{m} b_j - oj & & \quad (1.4)\\
\text{s.t.} \quad & \sum_{k=1}^{r} x_{ijk} = a_{ij} & & \forall i \in I, \quad \forall j \in J\\
& \sum_{l=1}^{p} y_{jlk} \leq 1 & & \forall j \in J, \quad \forall k \in K\\
& \sum_{j=1}^{m} y_{jlk} \leq 1 & & \forall l \in L, \quad \forall k \in K\\
& \sum_{j=1}^{m} x_{ijk} \leq 1 & & \forall i \in I, \quad \forall k \in K\\
& x_{ijk} \leq \sum_{l=1}^{p} y_{jlk} & & \forall i \in I, \quad \forall j \in J, \quad \forall k \in K\\
& \sum_{i=1}^{n} x_{ijk} \leq \sum_{l=1}^{p} y_{jkl} \times c_l & & \forall j \in J, \quad \forall k \in K\\
& y_{jlk} \leq t_{jk} & & \forall j \in J, \quad \forall l \in L, \quad \forall k \in K\\
& o_j \leq k + 4(1 - t_{jk}) & & \forall j \in J, \quad \forall k \in K\\
& b_j \geq k \times t_{jk} & & \forall j \in J, \quad \forall k \in K\\
& x_{ijk} \in \{0,1\} & & \forall i \in I, \quad \forall j \in J, \quad \forall k \in K\\
& y_{jlk} \in \{0,1\} & & \forall j \in J, \quad \forall l \in L, \quad \forall k \in K\\
& t_{jk} \in \{0,1\} & & \forall j \in J, \quad \forall k \in K\\
& o_j \in \mathbb{N} & & \forall j \in J\\
& b_j \in \mathbb{N} & & \forall j \in J
\end{aligned}$$

# 3
# Results

In this chapter we use two data sets received from the Christian College Groevenbeek that organizes the general open day. One data set consists of applicants for the morning and the other for applicants in the afternoon. The total amount of applicants in the morning is 599 and in the afternoon is 534. First, we will analyze both data sets. Then, we use the feasibility model to find the appropriate combination of rounds, lecture halls, and lecture hall capacities. Finally, we will add the objective functions to find the best possible schedules.

## 3.1. Data analysis

As already explained, teachers from several community colleges come to the open day to give presentations about studies provided by the community college they teach. There is a total of 32 different studies that are represented on this open day and there is a presentation about each study. From these 32 studies students were required to choose three. The data we received also consists of this information and we will use this information as input for our model.

From the data we can state that if we have 32 lecture halls available with infinite capacity it is enough to have only 3 rounds. In each round each student is able to follow one of the studies he or she has chosen. This is a simple solution, but not a realistic one. The building where the open day is held, does not have 32 lecture halls with a huge capacity. Additionally this results in a schedule that consists of a large number of presentations. To be exact, a total of $32 \times 3 = 96$ presentations. To create a good schedule we want as little presentations as possible, because this would lead to better use of the capacity of each lecture hall in each round. Besides that, the workload of each teacher would be minimized.

We can already state that 3 rounds and 10 available lecture halls with infinite capacity available for each round would not work, as this scenario would indicate we could only host 30 studies total while the aim is 32 studies within 3 rounds.

Also, 3 rounds and 11 available lecture halls with infinite capacity in each round would also not work, because of occurring cycles and the constraint that each student can only follow one presentation in each round. For example, say the following preferences of 5 students occur in the data set: ABC, ABD, ABE, and CDE. For the first three students we need to put A in round 1, B in round 2, and C, D, E in round 3. For the student with study choices CDE we need to have a lecture hall available in round 1 and round 2, so the student can also follow all of his or hers preferred choices, see Table 3.1. This means that we use two extra classrooms for two studies. For the other 27 sessions we still have 26 lecture halls available.

From the above we can conclude that if a cycle in the data occurs we definitely can not make a schedule when there are not enough lecture halls available. Notice that cycles will always occur in larger data sets, though this should not be an issue when the number of lecture halls suffices.

Table 3.1: Example of a part of a schedule including the study information sessions A, B, C, D, and E

| Round 1 | Round 2 | Round 3 |
|---|---|---|
| A | B | C |
| C | D | D |
| | | E |

For the open day we do not want a lot of rounds on both day parts, additionally we do not want to use too many lecture halls. The following scenario is set up for the purpose of determining the most efficient round division: each day part consists out of 4 hours, each round consists out of 50 minutes, and there are breaks in between each round to complete the 4 hour day part. So we want the schedule to consists of 4 rounds in each day part. Note that each round lasts an hour, thus each presentation on a study lasts an hour.

In the data we are given that each applicant chose 3 studies they prefer to acquire knowledge about. They had the option to choose from the 32 different studies offered, which are the same in the morning and afternoon. For the number of applicants for each study see Table 3.2.

Now if we look at Table 3.2 we can conclude that studies like "1. Dierverzorging", "17. Gaming/Webdesign", "27. Verzorgende/Verpleegkundige", "28. Pedagogisch werker", and "30. Sport en bewegen" have a high number of applicants. We can assume that the studies with the most applicants have the most influence on the amount of rounds. A regular lecture hall has a capacity of 30, so we can see that studies with more than 90 applicants are likely to be scheduled in each round. In the morning the studies with more than 90 applicants are: "1. Dierverzorging", "17. Gaming/Webdesign", "27. Verzorgende/Verpleegkundige", "28. Pedagogische werker", and "30. Sport en bewegen". In the afternoon the studies with more than 90 applicants are: "17. Gaming/Webdesign", "17. Verzorgende/Verpleegkundige", "28. Pedagogische werker", "30. Sport en bewegen", and "31. VEVA, Landmacht".

The study "17. Gaming/Webdesign" has 124 applicants in the afternoon, but if the capacity of each lecture hall is 30 we can already conclude that it is not possible to create a schedule that only consists of 4 rounds. So, the schedule for the afternoon consists of at least 5 rounds. We used a capacity of 30 in our previous argument, because a typical lecture hall in the building where the open day is held has a capacity of 30. One way to still be able to schedule all the applicants in 4 rounds is to have one lecture hall with a capacity more than 30, let us say 40. From a practical point of view, this is not an unrealistic assumption. In reality one bigger lecture is possible, because almost all educational institutions have a bigger lecture hall for meetings and events.

Another solution to the problem is to create an extra round for "17. Gaming/Webdesign" at the end of the afternoon for the students who attend the Christian College Groevenbeek. These students do not have to wait for the buses as they live nearby. For the teacher an extra round should not matter, as he or she is already scheduled for each round.

From the previous two solutions we can assume that using a bigger lecture hall is more efficient, because ultimately we do not want the extra round.

Besides studies with a lot of applicants we can see from Table 3.2 that there are studies with less than 30 applicants. For example: "6. Secretarieel", "18. Podiumtechniek", and "21. Operationele techniek". An ideal scenario would be to schedule these studies in just one round in both day parts, instead of in multiple rounds, which would results in almost empty lecture halls. Note that we do not enforce these decision, but that models (1.3) and (1.4) should automatically result in such a schedule.

Table 3.2: Offered studies at the open day and the corresponding number of applicants in the morning and afternoon

| | Studies | Morning | Afternoon |
|---|---|---|---|
| 1 | Dierverzorging | 115 | 73 |
| 2 | Bos en natuur | 34 | 36 |
| 3 | Outdoor en adventure | 52 | 41 |
| 4 | Bedrijfsadiministratie | 52 | 43 |
| 5 | Juridisch medewerker (sociaal of administratief) | 31 | 25 |
| 6 | Secretarieel | 28 | 17 |
| 7 | Detailhandel | 59 | 35 |
| 8 | Internationale handel en groothandel/IBS | 43 | 29 |
| 9 | Transport en logistiek | 22 | 41 |
| 10 | Horeca en facilitair leidinggevende | 63 | 71 |
| 11 | Bakkerijopleidingen | 31 | 53 |
| 12 | Luchtvaartdienstverlening | 33 | 25 |
| 13 | Toerisme en recreatie | 72 | 55 |
| 14 | Mode | 78 | 80 |
| 15 | Reclame, presentatie en communicatie | 40 | 40 |
| 16 | Grafische vormgeving | 70 | 52 |
| 17 | Gaming/Webdesign | 92 | 124 |
| 18 | Podiumtechniek | 14 | 21 |
| 19 | Automanagement | 43 | 38 |
| 20 | Middenkader engineering | 16 | 29 |
| 21 | Operationele techniek | 9 | 14 |
| 22 | Bouw-en infratechniek | 21 | 44 |
| 23 | Machinist & uitvoerder (grond/water/wegenbouw) | 13 | 15 |
| 24 | ICT (-beheer) | 43 | 50 |
| 25 | Laboratoriumopleidingen | 34 | 19 |
| 26 | Assistenten gezondheidszorg (apotheker /dokter / tandarts) | 35 | 41 |
| 27 | Verzorgende/Verpleegkundige | 106 | 102 |
| 28 | Pedagogisch werker | 119 | 92 |
| 29 | Kapper | 63 | 61 |
| 30 | Sport en bewegen | 120 | 102 |
| 31 | VEVA, Landmacht | 78 | 91 |
| 32 | Handhaving, toezicht en beveiliging | 48 | 43 |

## 3.2. RESULTS INTEGER LINEAR PROGRAMMING

In this section we will apply the mathematical models we formulated in Chapter 2 to the data sets we are given. We implemented the models in the optimization software AIMMS, which invokes the solver CPLEX to give a solution for our problem.

### 3.2.1. FEASIBILITY MODEL

To give recommendations about the number of of lecture halls and lecture hall capacities based on 4 rounds in each day part, we use the feasibility models (1.1) and (1.2) from Chapter 2. First, we consider lecture halls with unlimited capacities and calculate how many rounds we need for different number of lecture halls. From this we can determine what the minimum number of lecture halls is for 4 rounds and choose a practical number for further calculations.

Subsequently, we will determine what the impact is of limited capacity on the number of rounds. This allows us to determine a suitable capacity for each lecture hall. After we determined the right values we use this information together with models (1.3) and (1.4) to make a good quality schedule.

#### UNLIMITED CAPACITY

First, we consider an unrealistic scenario, namely lecture halls with unlimited capacity. By doing this we can determine the number of lecture halls we need for a schedule that consists of 4 rounds. From table 3.3 we can see how the number of rounds and lecture halls relate to each other when the capacity of each lecture hall is infinite.

Table 3.3: Minimum number of rounds calculated with model (1.2) for different number of lecture halls with unlimited capacity using data from applicants in the morning and afternoon

| Lecture halls | Rounds morning | Rounds afternoon |
|---|---|---|
| 5 | 9 | 9 |
| 10 | 5 | 5 |
| 11 | 5 | 5 |
| 12 | 5 | 5 |
| 13 | 4 | 4 |
| 14 | 4 | 4 |
| 15 | 4 | 4 |
| 20 | 4 | 4 |

You can see that the number of rounds decreases when the number of lecture halls increases. As expected, when we have more rounds we have less problems with occurring cycles. For example, say the following preferences of four students occur in the data set: ABC, CDE, ABD, and ABE. If there are three rounds we need to have a total of 7 presentations, and if there are four rounds we need to have a total of 6 presentations, see Tables 3.4 and 3.5. Therefore, more rounds will lead to fewer presentations as we can see in Table 3.3.

Table 3.4: Part of a schedule consisting of three rounds including the studies A, B, C, D, and E

| Round 1 | Round 2 | Round 3 |
|---|---|---|
| A | B | C |
| E | D | E |
| | | D |

Table 3.5: Part of a schedule consisting of four rounds including the studies A, B, C, D, and E

| Round 1 | Round 2 | Round 3 | Round 4 |
|---|---|---|---|
| A | B | C | D |
| | E | | E |

In Table 3.3 we calculated the minimum number of rounds for different number of lecture halls with infinite capacity. Note that we cannot find a feasible solution if we just have one or two rounds. Each student has to choose 3 studies, this means that it is not possible for a student to follow 3 studies when there are only one or two rounds available.

In Table 3.3 when we have 14 to 20 lecture halls available on both day parts we need to have a total of 4 rounds on both day parts. We choose to have 20 lecture halls available as this is a reasonable number.

#### LIMITED CAPACITY

We looked at scenarios with unlimited lecture hall capacities and concluded that we want to have 20 lecture halls available throughout the open day. Now we are going to look at more realistic scenarios by limiting the capacity of the lecture halls.

Let us say we only have small lecture halls available with a capacity of 10. In the morning the study "30. Sport en bewegen" has the highest number of applicants, which is a total of 120 applicants. If we only have lecture halls with a capacity of 10 we can conclude that we need a schedule with at least 12 rounds regardless of the number of available lecture halls. In the afternoon the study "17. Gaming/Webdesign" has the highest number of applicants, which is a total of 124 applicants. From this we can conclude that we need a schedule that has at least 13 rounds for different numbers of lecture halls. As you will probably notice this is not a realistic scenario, as we definitely have lecture halls available with capacities over 10 spots.

So what is a realistic number of capacity? To answer this question, a typical lecture hall in the building of Christian Groevenbeek College has a capacity of 30.

Table 3.6: Minimum number of rounds calculated with model (1.2) for different number of lecture halls with a capacity of 30 using data from applicants in the morning and afternoon

| Lecture halls | Rounds morning | Rounds afternoon |
|---|---|---|
| 5 | 15 | 16 |
| 10 | 8 | 9 |
| 15 | 5 | 6 |
| 20 | 4 | 5 |

If we look at Table 3.6 we see that in the afternoon we need at least 5 rounds if we have 20 lecture halls available with a capacity of 30. This is because in the afternoon the study "17. Gaming/Webdesign" has a total number of 124 applicants.

As previously mentioned we want a schedule where both day parts consists of 4 rounds. To achieve this we can use a lecture hall that has a larger capacity then 30. Let us say 40. Now model (1.2) gives us a minimum number of 4 rounds in the afternoon, and this is exactly what we want.

All in all, we think it is a realistic scenario to have 4 rounds, 19 lecture halls with a capacity of 30, and one lecture hall with a capacity of 40. Table 3.7 gives an overview of the results.

Table 3.7: Number of rounds, number of lecture halls and lecture hall capacities for the morning and afternoon.

| | Morning | Afternoon |
|---|---|---|
| Rounds | 4 | 4 |
| Classrooms | 20 | 20 |
| Classroom capacities | 19 lecture halls with capacity 30 and 1 lecture hall with capacity 40 | 19 lecture halls with capacity 30 and 1 lecture hall with capacity 40 |

### 3.2.2. OBJECTIVE FUNCTIONS

In this section we are going to create good quality schedules from the data we have received. To accomplish this we will use the integer linear programming models (1.3) and (1.4).

#### OBJECTIVE 1: MINIMIZE NUMBER OF PRESENTATIONS FOR EACH STUDY

If we look at the number of presentations the feasibility model we use gives us a total of 80 presentations in the morning and 80 presentations in the afternoon.

We do not want a lot of presentations as this increases the workload of each teacher. Besides, it does not improve the quality of the schedule. Let us say we only have lecture halls with a capacity of 30. To calculate the minimum number of presentations, we divide the total number of applicants for each study by 30 and round the resulting number up. Doing this for each study in Table 3.2 we get a total of at least 73 sessions in the morning and 70 in the afternoon. If we use one lecture hall with capacity 40 we can place one presentation in each round in this lecture hall. This means that theoretically the minimum number of presentations for 4 rounds is 69 in the morning and 66 in the afternoon.

For the data in the morning applying model (1.3) results in a schedule where the minimum number of presentations is 69. We can see that the theoretical minimum is achieved, which is quite remarkable. Apparently, we can divide the students in lecture halls in such a way that there are no problems of occurring cycles.

In the morning the studies "2. Bos en natuur: bloem, plant en groen", "11. Bakkerijopleidingen", "15. Reclame, presentatie en communicatie", and "26. Assistenten gezondheidszorg" are placed in the larger lecture hall in different rounds. Also, there is just one presentation for each of these studies in the morning. This is because the number of applicants is between the 30 and 40 and the capacity of the lecture hall they are placed is 40, see Tables 3.8 and 3.9.

| Studies | Round 1 | Round 2 | Round 3 | Round 4 |
|---|---|---|---|---|
| 2 | 1 | 1 | 0 | 0 |
| 11 | 1 | 1 | 0 | 0 |
| 15 | 1 | 0 | 0 | 1 |
| 26 | 0 | 1 | 1 | 0 |

Table 3.8: Rounds in which the studies "2. Bos en natuur: bloem, plant en groen", "11. Bakkerijopleidingen", "15. Reclame, presentatie en communicatie", and "26. Assistenten gezondheidszorg", are placed when applying the feasibility model (1.1) to the data in the morning

| Studies | Round 1 | Round 2 | Round 3 | Round 4 |
|---|---|---|---|---|
| 2 | 1 | 0 | 0 | 0 |
| 11 | 0 | 0 | 0 | 1 |
| 15 | 0 | 1 | 0 | 0 |
| 26 | 0 | 0 | 1 | 0 |

Table 3.9: Rounds in which the studies "2. Bos en natuur: bloem, plant en groen", "11. Bakkerijopleidingen", "15. Reclame, presentatie en communicatie", and "26. Assistenten gezondheidszorg", are placed when applying model (1.3) to the data in the morning

For the data in the afternoon applying model (1.3) results in a schedule where the minimum number of presentations is 66. We can see that also for the afternoon the theoretical minimum is achieved, which is again quite remarkable. Thus, the students are divided in lecture halls in such way that there are no problems of occurring cycles. In the afternoon the studies "17. Gaming/Webdesign", "19. Automanagement", "28. Pedagogische werker", and "31. VEVA, Landmacht" each are placed in the larger lecture hall in different rounds.

So what happens with the total workload of the teachers when the number of presentations is minimized? The total workload is the total number of rounds all teachers must be present at the college. This also includes gaps, which are rounds a teacher does not give a presentations. As previously mentioned, a round is an hour long, so we measure the total workload in hours. In this thesis we will also minimize the total workload using model (1.4).

When we minimize the number of presentations, the total workload is also minimized. The solution provided by the feasibility model (1.1) gives us a total workload of 93 hours in the morning and 90 hours in the afternoon. Minimizing the total number of presentations applying model (1.3) to the data sets gives us a total workload of 80 hours in the morning and afternoon, which is a decrease of 13 hours and 10 hours. Thus the number of gaps decreases as we minimize the number of presentations.

Table 3.10 gives an overview of the results obtained so far.

Table 3.10: Number of presentations and the total workload in the morning and afternoon when the feasibility model (1.1) and model (1.3) are applied to the data

| Morning | # Presentation | Workload |
|---|---|---|
| Feasibility Model | 80 | 93 |
| Objective 1 | 69 | 80 |
| Afteroon | # Presentation | Workload |
| Feasibility Model | 80 | 90 |
| Objective 1 | 66 | 80 |

For an overview of the timetables in the morning and the afternoon, see Tables 3.11 and 3.12.

Table 3.11: Timetable for the morning where the number of presentations is minimized. For each study you can find the number of students scheduled in each round. The number of students represented by a bold number are placed in the larger lecture hall. In the schedule the gaps are represented by cells filled with diagonal lines.

| Studies | Rounds | | | |
|---|---|---|---|---|
| | 1 | 2 | 3 | 4 |
| 1. Dierverzorging | 25 | 30 | 30 | 30 |
| 2. Bos en natuur | **34** | ■ | ■ | ■ |
| 3. Outdoor en adventure | ■ | ■ | 25 | 27 |
| 4. Bedrijfsadministratie | 22 | ▨ | 30 | ■ |
| 5. Juridisch medewerker | ■ | 15 | ▨ | 16 |
| 6. Secretarieel | ■ | ■ | ■ | 28 |
| 7. Detailhandel | 29 | 30 | ■ | ■ |
| 8. Internationale handel | 13 | ▨ | ▨ | 30 |
| 9. Transport en logistiek | ■ | ■ | 22 | ■ |
| 10. Horeca en facilitair leidinggevende | 16 | ▨ | 24 | 23 |
| 11. Bakkerijopleidingen | ■ | ■ | ■ | **31** |
| 12. Luchtvaartdienstverlening | 10 | ▨ | 23 | ■ |
| 13. Toerisme en recreatie | 24 | 21 | 27 | ■ |
| 14. Mode | 27 | 27 | ▨ | 24 |
| 15. Reclame, presentatie en communicatie | ■ | **40** | ■ | ■ |
| 16. Grafische vormgeving | ■ | 22 | 26 | 22 |
| 17. Gaming/Webdesign | 22 | 22 | 19 | 29 |
| 18. Podiumtechniek | ■ | ■ | ■ | 14 |
| 19. Automanagement | 29 | ▨ | ▨ | 14 |
| 20. Middenkader engineering | ■ | 16 | ■ | ■ |
| 21. Operationele techniek | ■ | ■ | 9 | ■ |
| 22. Bouw-en infratechniek | 21 | ■ | ■ | ■ |
| 23. Machinist en uitvoerder | ■ | 13 | ■ | ■ |
| 24. ICT (-beheer) | 22 | 21 | ■ | ■ |
| 25. Laboratoriumopledingen | ■ | 16 | 18 | ■ |
| 26. Assistenten gezondheidszorg | ■ | ■ | **35** | ■ |
| 27. Verzorgende/Verpleegkundige | 28 | 29 | 20 | 29 |
| 28. Pedagogisch medewerker | 29 | 30 | 30 | 30 |
| 29. Kapper | 28 | ▨ | 19 | 16 |
| 30. Sport en bewegen | 30 | 30 | 30 | 30 |
| 31. VEVA, Landmacht | ■ | 27 | 21 | 30 |
| 32. Handhaving, toezicht en beveiliging | ■ | 30 | ▨ | 18 |

Table 3.12: Timetable for the afternoon where the number of presentations is minimized. For each study you can find the number of students scheduled in each round. The number of students represented by a bold number are placed in the larger lecture hall. In the schedule the gaps are represented by cells filled with diagonal lines.

| Studies Rounds | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1. Dierverzorging | 23 | 24 | ■ | 26 |
| 2. Bos en natuur | 23 | 13 | ■ | ■ |
| 3. Outdoor en adventure | ■ | 17 | 24 | ■ |
| 4. Bedrijfsadministratie | 18 | 25 | ■ | ■ |
| 5. Juridisch medewerker | ■ | 25 | ■ | ■ |
| 6. Secretarieel | ■ | ■ | 17 | ■ |
| 7. Detailhandel | ■ | ■ | 15 | 20 |
| 8. Internationale handel | 29 | ■ | ■ | ■ |
| 9. Transport en logistiek | ■ | 13 | ▨ | 28 |
| 10. Horeca en facilitair leidinggevende | 11 | ▨ | 30 | 30 |
| 11. Bakkerijopleidingen | 29 | ▨ | ▨ | 24 |
| 12. Luchtvaartdienstverlening | ■ | 25 | ■ | ■ |
| 13. Toerisme en recreatie | 29 | ▨ | ▨ | 26 |
| 14. Mode | 26 | 29 | ▨ | 25 |
| 15. Reclame, presentatie en communicatie | 22 | ▨ | 18 | ■ |
| 16. Grafische vormgeving | ■ | 26 | ▨ | 25 |
| 17. Gaming/Webdesign | 30 | 30 | 30 | **34** |
| 18. Podiumtechniek | ■ | 22 | ■ | ■ |
| 19. Automanagement | ■ | ■ | **38** | ■ |
| 20. Middenkader engineering | 29 | ■ | ■ | ■ |
| 21. Operationele techniek | ■ | ■ | ■ | 14 |
| 22. Bouw-en infratechniek | ■ | 15 | 29 | ■ |
| 23. Machinist en uitvoerder | ■ | 15 | ■ | ■ |
| 24. ICT (-beheer) | ■ | 22 | ▨ | 28 |
| 25. Laboratoriumopledingen | ■ | ■ | ■ | 19 |
| 26. Assistenten gezondheidszorg | 30 | ▨ | ▨ | 11 |
| 27. Verzorgende/Verpleegkundige | 18 | 30 | 28 | 26 |
| 28. Pedagogisch medewerker | ■ | **35** | 29 | 28 |
| 29. Kapper | 15 | 18 | 28 | ■ |
| 30. Sport en bewegen | 27 | 22 | 27 | 26 |
| 31. VEVA, Landmacht | **35** | 30 | ▨ | 26 |
| 32. Handhaving, toezicht en beveiliging | 21 | 22 | ■ | ■ |

### OBJECTIVE 2: MINIMIZE WORKLOAD FOR EACH TEACHER

The total workload is the total number of rounds all teachers must be present at the college. This also includes gaps, which are rounds a teacher does not give a presentation, but needs to be present. It is mentioned before that a round is an hour long, so we measure the total workload in hours.

If we look at the total workload the feasibility model gives us a total of 93 hours in the morning and 90 hours in the afternoon. Minimizing the total workload using model (1.4) gives us a total workload of 69 hours in the morning and 66 hours in the afternoon. Besides that, the number of presentations drops from 80 to 69 in the morning and 80 to 66 in the afternoon. Surprisingly, the number of presentations does not increase. The number of presentations on both day parts is the same as the theoretical minimum. Thus, the schedules contain zero gaps as the workload is the same as the minimum number of presentations on both day parts.

In Table 3.13 we combined the data of both day parts to get a better overview of the results. When we minimize the total workload of both day parts separate this results in schedules that contain zero gaps and where the total number of presentations is equal to the theoretical minimum. But if we put both day parts together the schedule still contains gaps between the two day parts for a lot of studies, which we do not want, see Table 3.13. Thus, it is better to apply model (1.4) on both day parts combined to improve the schedule.

Table 3.13: Timetable for the morning and afternoon where the total workload is minimized for each day part separate. For each study the number of students scheduled in each round is given. Rounds 1 till 4 are in the morning and rounds 5 till 8 are in the afternoon. In the schedule the gaps are represented by cells with diagonal lines.

| Studies | Rounds | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|---|
| 1. Dierverzorging | | 27 | 30 | 28 | 30 | 29 | 25 | 19 | ■ |
| 2. Bos en natuur | | ■ | 34 | ▨ | ▨ | 17 | 19 | ■ | ■ |
| 3. Outdoor en adventure | | 30 | 22 | ▨ | ▨ | 27 | 14 | ■ | ■ |
| 4. Bedrijfsadministratie | | 25 | 27 | ▨ | ▨ | ▨ | ▨ | 19 | 24 |
| 5. Juridisch medewerker | | ■ | ■ | 13 | 18 | ▨ | 25 | ■ | ■ |
| 6. Secretarieel | | ■ | ■ | ■ | 28 | ▨ | ▨ | 17 | ■ |
| 7. Detailhandel | | 30 | 29 | ▨ | ▨ | ▨ | 23 | 12 | ■ |
| 8. Internationale handel | | ■ | 17 | 26 | ▨ | 29 | ■ | ■ | ■ |
| 9. Transport en logistiek | | ■ | ■ | ■ | 22 | 21 | 20 | ■ | ■ |
| 10. Horeca en facilitair leidinggevende | | ■ | ■ | 35 | 28 | ▨ | 20 | 27 | 24 |
| 11. Bakkerijopleidingen | | 17 | 14 | ▨ | ▨ | ▨ | 25 | 28 | ■ |
| 12. Luchtvaartdienstverlening | | ■ | ■ | 9 | 24 | 25 | ■ | ■ | ■ |
| 13. Toerisme en recreatie | | 29 | 18 | 25 | ▨ | 26 | 29 | ■ | ■ |
| 14. Mode | | 23 | 26 | 29 | ▨ | ▨ | 23 | 27 | 30 |
| 15. Reclame, presentatie en communicatie | | 15 | 25 | ▨ | ▨ | ▨ | ▨ | 19 | 21 |
| 16. Grafische vormgeving | | 20 | 24 | 26 | ▨ | ▨ | ▨ | 25 | 27 |
| 17. Gaming/Webdesign | | ■ | 27 | 30 | 35 | 27 | 38 | 29 | 30 |
| 18. Podiumtechniek | | ■ | ■ | 14 | ▨ | ▨ | ▨ | ▨ | 21 |
| 19. Automanagement | | ■ | 26 | 17 | ▨ | ▨ | ▨ | 20 | 18 |
| 20. Middenkader engineering | | 16 | ▨ | ▨ | ▨ | 29 | ■ | ■ | ■ |
| 21. Operationele techniek | | ■ | ■ | ■ | 9 | ▨ | ▨ | ▨ | 14 |
| 22. Bouw-en infratechniek | | ■ | ■ | 21 | ▨ | ▨ | 24 | 20 | ■ |
| 23. Machinist en uitvoerder | | 13 | ▨ | ▨ | ▨ | ▨ | ▨ | 15 | ■ |
| 24. ICT (-beheer) | | ■ | ■ | 20 | 23 | ▨ | 30 | 20 | ■ |
| 25. Laboratoriumopleidingen | | 34 | ▨ | ▨ | ▨ | ▨ | 19 | ■ | ■ |
| 26. Assistenten gezondheidszorg | | ■ | 16 | 19 | ▨ | 24 | 17 | ■ | ■ |
| 27. Verzorgende/Verpleegkundige | | 30 | 26 | 20 | 30 | 29 | 22 | 21 | 30 |
| 28. Pedagogisch medewerker | | 30 | 30 | 29 | 30 | ▨ | 28 | 24 | 40 |
| 29. Kapper | | ■ | 19 | 15 | 29 | ▨ | ▨ | 32 | 29 |
| 30. Sport en bewegen | | 30 | 30 | 30 | 30 | 30 | 29 | 15 | 28 |
| 31. VEVA, Landmacht | | 29 | 20 | 29 | ▨ | 36 | 28 | 27 | ■ |
| 32. Handhaving, toezicht en beveiliging | | ■ | 22 | 26 | ▨ | ▨ | 14 | 29 | ■ |

### OBJECTIVE 2: MORNING AND AFTERNOON COMBINED

Beforehand, we solved the open days scheduling problem for both day parts separate. This resulted in a good schedule when we minimized the total workload. As we solved the problem separate, this did not result in a good schedule when we combine the schedules of both day parts, see Table 3.13.

Because we have 4 rounds on each day part teachers who give presentations on studies with a lot of applicants give presentations in each round in each day part. For example, studies like "27. Verzorgende/ Verpleegkundige", "30. Sport en bewegen", see Table 3.13. On the other hand, teachers with presentations on a study with not many applicants will give one, two or three presentations in each day part. For these teachers we do not want these presentations to be at the beginning of the morning and at the end of the afternoon as this can result in multiple gaps in their schedule. For example, the studies "18. Podiumtechniek" and "6. Secretarieel", see Table 3.13. Instead, we want them to be at the end of the morning and at the beginning of the afternoon. To achieve this we need to solve the open days scheduling problem for both day parts combined and minimize the total workload applying model (1.4) to the combined data. We combine the solutions found by applying model (1.4) to both data sets separate to provide the solver CPLEX with an initial solution. This will improve the CPLEX solve process, as already explained in Section 1.6.

Table 3.14: Timetable for the morning and afternoon where the total workload is minimized applying model (1.4) to the data sets combined. For each study the number of students scheduled in each round is given. Rounds 1 till 4 are in the morning and rounds 5 till 8 are in the afternoon. In the schedule the gaps are represented by cells with diagonal lines.

| Studies Rounds | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| 1. Dierverzorging | 30 | 30 | 27 | 28 | 29 | ▨ | 30 | 14 |
| 2. Bos en natuur | ■ | 28 | 6 | ▨ | ▨ | ▨ | 6 | 30 |
| 3. Outdoor en adventure | 30 | 22 | ▨ | ▨ | 19 | 22 | ■ | ■ |
| 4. Bedrijfsadministratie | 28 | 24 | ▨ | ▨ | 22 | ▨ | 21 | ■ |
| 5. Juridisch medewerker | 15 | 16 | ▨ | ▨ | ▨ | 25 | ■ | ■ |
| 6. Secretarieel | ■ | ■ | ■ | 28 | ▨ | 4 | 13 | ■ |
| 7. Detailhandel | ■ | 29 | 30 | ▨ | 35 | ■ | ■ | ■ |
| 8. Internationale handel | ■ | ■ | 29 | 14 | 15 | 14 | ■ | ■ |
| 9. Transport en logistiek | 22 | ▨ | ▨ | ▨ | ▨ | 21 | 20 | ■ |
| 10. Horeca en facilitair leidinggevende | ■ | ■ | 30 | 33 | 27 | 28 | 16 | ■ |
| 11. Bakkerijopleidingen | ■ | ■ | 17 | 14 | 24 | ▨ | 29 | ■ |
| 12. Luchtvaartdienstverlening | ■ | ■ | 6 | 27 | 25 | ■ | ■ | ■ |
| 13. Toerisme en recreatie | 30 | 13 | 29 | ▨ | ▨ | 26 | 29 | ■ |
| 14. Mode | ■ | 30 | 18 | 30 | ▨ | 29 | 21 | 30 |
| 15. Reclame, presentatie en communicatie | ■ | ■ | 26 | 14 | ▨ | ▨ | 26 | 14 |
| 16. Grafische vormgeving | ■ | 30 | 17 | 23 | 25 | 27 | ■ | ■ |
| 17. Gaming/Webdesign | ■ | 32 | 30 | 30 | 25 | 39 | 30 | 30 |
| 18. Podiumtechniek | ■ | ■ | ■ | 14 | 21 | ■ | ■ | ■ |
| 19. Automanagement | ■ | ■ | 25 | 18 | 26 | 12 | ■ | ■ |
| 20. Middenkader engineering | ■ | 16 | ▨ | ▨ | ▨ | ▨ | 29 | ■ |
| 21. Operationele techniek | ■ | ■ | ■ | 9 | ▨ | 14 | ■ | ■ |
| 22. Bouw-en infratechniek | ■ | ■ | ■ | 21 | 16 | 28 | ■ | ■ |
| 23. Machinist en uitvoerder | ■ | 13 | ▨ | ▨ | 15 | ■ | ■ | ■ |
| 24. ICT (-beheer) | ■ | 13 | 30 | ▨ | 29 | 21 | ■ | ■ |
| 25. Laboratoriumopleidingen | 20 | 14 | ▨ | ▨ | ▨ | 19 | ■ | ■ |
| 26. Assistenten gezondheidszorg | ■ | 6 | 29 | ▨ | 25 | 16 | ■ | ■ |
| 27. Verzorgende/Verpleegkundige | 29 | 30 | 19 | 28 | 29 | 25 | 21 | 27 |
| 28. Pedagogisch medewerker | 29 | 30 | 30 | 30 | ▨ | 30 | 32 | 30 |
| 29. Kapper | ■ | ■ | 38 | 25 | 24 | ▨ | 9 | 28 |
| 30. Sport en bewegen | 40 | 27 | 23 | 30 | 25 | 27 | 20 | 30 |
| 31. VEVA, Landmacht | ■ | 27 | 30 | 21 | ▨ | 21 | 30 | 40 |
| 32. Handhaving, toezicht en beveiliging | ■ | 25 | ▨ | 23 | 18 | ▨ | 25 | ■ |

The solution we have found is given in Table 3.14. When we solved the problem for both day parts separate this resulted in a combined schedule with a total workload of 196 hours, see Table 3.13. When we minimize the total workload for both day parts combined the total workload is equal to 179 hours, see Table 3.14. We have minimized the total workload with 17 hours, which is quite good. The number of gaps in the schedule has been reduced from 61 gaps to 41 gaps, which is a decrease of a total of 20 gaps. Thus, we have improved the quality of the schedule just as we wanted.

## 3.3. DISCUSSION RESULTS INTEGER LINEAR PROGRAMMING

In the previous sections we have applied both objective functions from Chapter 2 on the data sets we received. The data consisted of two sets: one for applicants for the morning and one for applicants for the afternoon. First, we solved the open days scheduling problem for both day parts separate. For an overview of the results, see Table 3.15.

Table 3.15: Number of presentations and the total workload in the morning and afternoon for the feasibility model, objective 1 and objective 2

| Morning | # Presentation | Workload |
|---|---|---|
| Feasibility Model | 80 | 93 |
| Objective 1 | 69 | 80 |
| Objective 2 | 69 | 69 |
| **Afteroon** | **# Presentation** | **Workload** |
| Feasibility Model | 80 | 90 |
| Objective 1 | 66 | 80 |
| Objective 2 | 66 | 66 |

From Table 3.15 we can conclude that objective 2 results in a better schedule applied separately on both day parts. Like we have already seen in Table 3.13, this does not improve the quality of the schedule as it contains gaps when we combine the schedules of both day parts. To improve this we combined the solutions found by applying model (1.4) to the sets of data to provide the solver CPLEX with an initial solution. Doing so improved the quality of the schedule, see Table 3.14.

Besides the problem of gaps in the schedule, another problem is the division of applicants over lecture halls. For further research we would recommend to look into this, as this problem is quite interesting. When we apply model (1.4) to the data sets combined this results in multiple studies where the spread of the applicants over lecture halls is not preferable. In Table 3.16 we have listed some of these studies. For example, the study "2. Bos en natuur" takes place in rounds two, three, seven, and eight. In the second round the number of students to attend the presentation is 28, in the third and seventh round this number is 6, and in the eighth round this number is 30. This division of the number of applicants for the study is not preferable as in the third and seventh round it results in an almost empty lecture hall. We do not want this, because a teacher would rather give a presentation to a lecture hall that is full than a lecture hall that is almost empty. Eventually we want the spread of the applicants over the lecture halls to be somewhat equally divided. One way to spread the number of applicants evenly is to iteratively add constraints. These constraints are only added to studies where the division of applicants over the rounds is not preferable.

Table 3.16: Studies where the spread of applicants over the rounds is not preferable. For each study the number of students in each round is given.

| Studies Rounds | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| 2. Bos en natuur | - | 28 | 6 | - | - | - | 6 | 30 |
| 6. Secretarieel | - | - | - | 28 | - | 4 | 13 | - |
| 21. Operationele techniek | - | - | - | 9 | - | 14 | - | - |
| 26. Assistenten gezondheidszorg | - | 6 | 29 | - | 25 | 16 | - | - |

# 4

# Conclusion, discussion, and recommendations

## 4.1. Conclusion

In this thesis we have addressed a timetabling problem for the open day at the Christian College Groevenbeek. This open day consists of two separate day parts: the morning and the afternoon. For both day parts we have received a data set. These data sets consist of students and their preferred studies.

First, we solved the timetabling problem separate for both day parts. To solve the problem we created several mathematical models formulated as an ILP. After that, we implemented these models in AIMMS to solve them for the data sets we have received.

The first model we created was the feasibility model (1.1). We used this model to determine the appropriate number of lecture halls and lecture hall capacities when we have 4 rounds on both day parts. To achieve 4 rounds on both day parts we found that the best combination to have is 19 lecture halls with a capacity of 30 and 1 lecture hall with a capacity of 40.

In addition, for a good quality schedule objective functions were considered. The first objective was to minimize the number of presentations. The second objective was to minimize the total workload. We added each objective and their corresponding constraints to model (1.1), which resulted in models (1.3) and (1.4). After applying both models (1.3) and (1.4) to the data sets we concluded that the second objective resulted in a better schedule, as it achieves the theoretical minimum number of presentations and creates zero gaps in the schedules for both day parts.

When we combined the schedules for both day parts this did not result in a good schedule as some studies still contained gaps between the two day parts, see Table 3.13. To improve this we applied model (1.4) for both data sets combined, see Table 3.14.

## 4.2. Discussion and recommendations for further research

A big problem in this thesis was the time it took to find an optimal solution with model (1.4) using AIMMS. Especially, when we combined the data sets we had a big integrality gap. Therefore, it was very time consuming to find an optimal schedule. As already explained, a strong LP relaxation results in an integrality gap close to one. We can conclude that our LP relaxation was not strong enough and for further research it is recommended to impose more valid inequalities to model (1.4). This will result in a stronger LP relaxation, which will speed up the running time of the program.

In Section 3.2.2, we showed that in the schedules the division of applicants over lecture halls is not preferable for some studies. For further research we definitely recommend to look into this. One way to solve this problem is to iteratively impose constraints to model (1.4) based on a solution the model has given us. These constraints are added to studies where the division of applicants over the rounds is not preferable.

As already explained, the timetabling problem is not an easy problem. There is definitely still a lot of research that remains to be done on this topic. I hope this thesis will contribute to existing research and gave insights into future research possibilities.

# BIBLIOGRAPHY

[1] T. B. Cooper and J. H. Kingston, *The complexity of timetable construction problems,* In the Practice and Theory of Automated Timetabling, pp 283-295, (1996) .

[2] A. Wren, *Scheduling, timetabling and rostering – a special relationship?* The Practice and Theory of Automated Timetabling I, 1153, pp 46-75, (1996) .

[3] A. Schaerf, *A survey of automated timetabling,* Artificial Intelligence Review, 13(2), pp 87-127, (1999) .

[4] M. W. Carter, G. Laporte, and S. Lee, *Examination timetabling: Algorithmic strategies and applications,* Journal of the Operational Research Society, 47(3), pp 373-383, (1996) .

[5] D. Werra, *An introduction to timetabling,* European Journal of Operational Research, 19, pp 151-162, (1985) .

[6] S. Abdullah, *Heuristic approaches for university timetabling problems*, Ph.D. thesis, University of Nottingham (2006).

[7] B. Esfahbod, *P versus np problem,* Retrieved from https://commons.wikimedia.org/wiki/File:P_np_np-complete_np-hard.svg, (2007) .

[8] S. Rebennack, *Ellipsoid method,* Encyclopedia of Optimization, 2, pp 890-899, (2008) .

[9] R. Robere, *Interior Point Methods and Linear Programming*, Ph.D. thesis, University of Toronto (2012).

[10] C. H. Papadimitriou and K. Steiglitz, *Combinatorial optimization; algorithms and complexity,* Prentice-Hall, Inc., Upper Saddle River, NJ, USA, (1982) .

[11] S. Albert, *Solving mixed integer linear programs using branch and cut algorithm,* BSc thesis, North Carolina State University (2004) .

[12] *Gurobi Optimizer Reference Manual*, Gurobi Optimization, 6th ed.

[13] *User's Manual for CPLEX*, International Business Machines Corporation, 12th ed.

[14] H. W. Kuhn, *The hungarian method for the assignment problem,* Naval Research Logistics Quarterly, 2, pp 83-97, (1955) .

[15] H. P. Williams, *The reformulation of two mixed integer programming models,* Mathematical Programming, 14, pp 31-325, (1978) .