The Collaborative Traineeship Planning Problem

FEM21030-18[1]

Author: Menno van Beek BSc.
412261

Supervisor: dr. K.S. Postek
Second assessor: dr. S. Ağralı

March, 2020

[1]Erasmus University Rotterdam, Erasmus School of Economics, Master's program Econometrics and Management Science specialization in Operations Research and Quantitative Logistics.

Abstract

In this thesis we study the collaborative traineeship planning problem (CTPP). This problem consists of the construction of an initial traineeship planning and the rescheduling of an existing planning in case of disturbances. We start by pointing out some places in the literature related to the CTPP. We then formulate the initial planning problem as a Mixed Integer Programming problem (MIP). Thereafter, we formulate the rescheduling problem as a MIP by extending the MIP formulation of the initial planning problem. We prove both problems are $\mathcal{NP}$-complete. Next, we elaborate on an exact approach for the initial planning problem wherein we exploit variable fixing methods and valid inequalities to increase the performance of a branch-and-bound algorithm. Furthermore, we consider a three-stage genetic algorithm as heuristic approach to solve the initial planning problem. We formulate the subproblem in this genetic algorithm as a MIP and impose symmetry breaking constraints in an attempt to improve the performance in terms of computation time. Also, for the rescheduling problem we consider an exact and a heuristic approach. In the exact approach we again take advantage of variable fixing methods and valid inequalities. The heuristic approach consists of a variable neighborhood search strategy, where each neighborhood is evaluated by means of a tabu search procedure. Finally, we provide results for both the initial planning as well as the rescheduling problem.

Keywords: Collaborative Traineeship Planning, Initial Planning Problem, Rescheduling Problem, Staffing Targets, Genetic Algorithm, Tabu Search, Complexity Analysis.

Preface

What you are about to read is a thesis submitted to obtain the Master of Science degree in the field of Econometrics, given the grade g of this thesis satisfies

$$5.5 \leq g \leq 10.$$

I would like to take the opportunity to express my gratitude to dr. Krzysztof Postek for being my supervisor throughout this process. His guidance and theoretical insights have greatly helped to shape this thesis. Moreover, a special thanks to dr. Semra Ağralı for being the second reader of this thesis. Furthermore, I would like show my appreciation to Gregor Brandt and Mark van der Goot for their valuable suggestions during my graduation internship at QuO Mare. Besides the above mentioned people, the ping-pong matches at QuO Mare against Jaron also helped a bit.

You will notice that in this thesis, I refer to 'we' every time I make a statement. This is not caused by the author being socially embarrassed or whatsoever. It is meant to engage you, the reader, into the development process. To you as a reader, I would like to express my appreciation. You almost succeeded in reading at least one page of my master thesis. I hope you will enjoy reading the rest.

Contents

List of Figures

List of Tables

List of Algorithms

1 Introduction

Motivated by increasing labor costs, companies are looking for opportunities to improve the productivity of their employees. Among the available options, a common strategy is to train employees in training programs (also known as traineeships). Within such a program, employees are trained in multiple business units to increase their experience level which is beneficial in terms of productivity. At the same time the business units themselves also benefit from being fulfilled. In other words, there is a common interest for employees and business units to make an efficient collaborative traineeship planning. Creating such a planning is considered a non-trivial task as multiple objectives and constraints on the traineeship program must be taken into account. Thereby, whenever disruptive events occur the planning can be modified with respect to the existing planning to deal with these disturbances.

In this research we focus on the Collaborative Traineeship Planning Problem (CTPP). This problem consists of: (*i*) constructing an initial traineeship planning, and (*ii*) rescheduling an existing planning in case of disruptive events. We aim to develop an algorithm (or two algorithms) to solve the initial planning and rescheduling problem. Hence, the main focus of this thesis is to address the following research question:

Can we design an algorithm that produces a collaborative traineeship planning within a reasonable amount of time while satisfying constraints on the traineeship program such that it is flexible in the trade-off of multiple objectives?

1.1 Contribution and Structure of this Thesis

We will address this research question as follows. First, we provide an extensive description of the aspects of this problem and review the corresponding literature. Thereafter, our first contribution to the existing literature is a Mixed Integer Programming (MIP) formulation of the initial planning variant of the CTPP. This MIP formulation is then extended to formulate the rescheduling variant of the CTPP. Thereby, we prove both problems are $\mathcal{NP}$-complete. Next, we develop both an exact and a heuristic approach to solve the initial planning problem. In the exact approach we exploit variable fixing methods and valid inequalities. In addition to this, we propose a three-stage genetic algorithm as heuristic approach. For the rescheduling problem we consider an exact approach where we again take advantage of variable fixing methods and valid inequalities. Next to that, we consider a tabu search algorithm with multiple neighborhoods to solve the rescheduling problem.

This thesis is structured as follows. We start in Section 2 with a comprehensive description of the CTPP. Next, in Section 3 we present a review of the existing literature related to the problem. In Section 4 we elaborate on the data required for this problem. Thereafter, we formulate both variants of the problem in Section 5 as a MIP and proof both variants of the problem are $\mathcal{NP}$-complete. In Section 6 we describe the methodology to solve both problems. We discuss the results in Section 7. Lastly, we conclude this thesis in Section 8.

2 Problem Description

For companies it is becoming increasingly more common to train employees in collaborative training programs. An example of a collaborative training program is a group of people (employees) that follow a certain training program in which different business units must be involved, but wherein the components themselves also benefit from being fulfilled. In this case we consider both a common as well as an individual interest in the training program, which can make planning more complex. In current literature the collaborative traineeship aspect in the optimization of crew planning is not yet addressed. The problem is even more challenging if a rolling horizon is considered, in which the crew is rescheduled whenever disruptions occur or new crew enters the training program. Combined with specific restrictions on the training program, multiple objectives and structure on the internships, the problem becomes even more complicated from both a scientific point of view as well as from a practical perspective. To grasp the complexity of this problem in more detail we provide a description of each aspect below.

Collaborative traineeship planning. The collaborative traineeship aspect consists of both the common and individual interest. The individual interest refers to, on the one hand, people that want to complete the training program and, on the other hand, the interest of the company to have the internships staffed. The common interest aspect refers to the fact that both entities have jointly interest in the internships, each from their own perspective. Hence, the collaborative characteristic is a major extension to regular crew planning problems and of utmost importance to this problem.

Consecutive training program. From a practical perspective the schedule for each person in the training program should consist of consecutively planned internships. That is, for each person the training program shall consists of planned internships without gaps in-between. The reason for this is to ensure the training program is completed by each person within a specified time period. Furthermore, this is also supported by the practical incentive related to high cost for people not being scheduled.

Internship and location structure. To allow for more flexibility we assume the number of internships available at each location can vary to some extend. For each internship a soft target on the minimum and maximum number this internship is available at each location is set by the company. Additional costs are incurred in the case the target is violated. We refer to a violation of the minimum (maximum) number of available internships as understaffing (overstaffing). These costs are made to hire extra people (in case of understaffing) or taking care of additional guidance (in case of overstaffing). The reason for this flexibility is that we already restricted that each person should have a consecutive training program. That is, the problem cannot be restricted in both the individual and common interest. Furthermore, to enhance even more flexibility we allow some variation in the the duration of each internship per person. For each internship a range on the duration per person is provided, representing the minimum and maximum duration that person shall perform the internship before it is considered as completed.

Block-wise structure. Each internship has a corresponding block, which forces structure on the order in which internships can be performed. A block is closely related to experience level, but represents a more generic framework. The block structure imposes that a person can perform internships of a certain block if that person completed the internships of the prior block(s). That is, we assume all internships corresponding to a certain block can be performed in an arbitrary order. However, the order in which two internships are performed that correspond to two distinct blocks is fixed in advance. This block structure is required for the training program to make sure that the experience level of an person is matched to the difficulty of the internship.

Rolling horizon and rescheduling. Whenever new people enter the training program, the current schedule is insufficient and therefore has to be rescheduled. That is, new people enter the program in a rolling horizon and a new schedule has to be made with respect to the current schedule. The current schedule is respected in the rescheduling to ensure the stability of the schedule. For example, if a person already arranged peripheral matters for a certain internship it is undesirable to assign another internship (or location) to this person. This aspect is especially important for internships scheduled in the near future and becomes decreasingly important further in the future.

Multi-objective. The previously described aspects and related objectives show that this problem is has multiple objectives. That is, the objectives are maximizing the number of people that complete the training program, minimizing the total cost corresponding to over- and understaffing internships and minimizing the rescheduling penalty whenever disruptions occur. To incorporate each objective appropriately, some thought need to be given on the trade-off between the objectives. That is, which objectives are most important and how each objective is taken into account.

3 Literature Review

In this section we elaborate on literature related to crew scheduling problems. We start with an general overview of regular crew scheduling problems. Thereafter, we discuss literature that focuses on crew rescheduling. Lastly, we review literature that is specifically related to the collaborative traineeship planning problem.

3.1 Crew Scheduling Problems

In the last few decades increasingly more crew scheduling problems are studied, mainly motivated by economic considerations to reduce labor costs. One of the first crew scheduling problems is introduced by Edie (1954) and Dantzig (1954) who investigate crew scheduling at traffic toll booths using a Linear Programming approach. Since then, various crew scheduling problems and extensions have been considered. Most of the scientific literature focuses on real-world applications of crew scheduling problems, among which the airline industry is one well-known example of crew optimization. An illustration of various constraints and objectives provided by the airline industry in crew optimization is presented in Kohl and Karisch (2004). Another illustration of crew scheduling in practice is given in Billionnet (1999), wherein a regular crew scheduling problem with skilled workers is considered.

In some applications, a general planning problem is solved sequentially. That is, a problem is split into smaller pieces that are solved sequentially, for example in the airline industry the problem of flight and crew scheduling is solved sequentially. In these cases the solution to the crew scheduling problem is related to the assignment of aircraft to flights. Hence, solving these problems simultaneously rather then sequentially could lead to better solutions. In Haase et al. (2001) an integrated approach to the vehicle and crew scheduling problem is presented and solved by a column generation approach for the crew schedules. This approach is integrated in a branch-and-bound algorithm where the side constraints on the vehicles guarantee that an optimal vehicle assignment can be derived afterwards in polynomial time. A second example of integrating vehicle and crew optimization is given in Huisman and Wagelmans (2006), wherein a dynamic variant of the problem is considered. That is, a schedule is not constructed for the whole time horizon, but is generated dynamically, for example the schedule can be generated every hour for the next hour. Their approach has the advantage that the assumption that travel times are only assumed to be known for the next hour rather than for the whole time period (in contrast to regular vehicle scheduling). Another example of a simultaneous approach is described in Cordeau et al. (2001) where a Benders decomposition is studied to solve the integrated problem of aircraft routing and crew scheduling. The Benders decomposition is used in order to handle linking constraints between aircraft and crew. In this approach the Benders master- and subproblem consists of aircraft routing and crew pairing, respectively. A trade-off in decomposition techniques is described in Beliën and Demeulemeester (2007).

Another topic is incorporating the preference of employees. For example, using a column generation approach, a multi-objective problem is considered in Bard and Purnomo (2005), wherein the preferences of nurses are taken into account. The objective coefficients are determined by the degree to which individual preferences are violated. The problem is solved by column generation in combination with integer programming and heuristics. In Hanne et al. (2009) another multi-objective problem is considered wherein the modelling is discussed including the treatment of constraints, consideration of preferences and formulation of several objective functions. Lastly, in Rasmussen et al. (2012) home care crew scheduling is considered in which the preferences of home care crew are clustered to take the soft preferences constraints into account.

Other extensions to regular crew scheduling problems are a stochastic approach to protect against delays in the airline industry in Yen and Birge (2006) and a weighted cost function in a multi-objective nurse scheduling problem in Parr and Thompson (2007). In Brunner and Edenharter (2011) different experience levels of employees are considered in a long term scheduling problem. Lastly, flexible employee availability is studied in Ağralı et al. (2017). Although most of the solution approaches exploit column generation, alternative solution approaches have also been considered. Two examples are the simulated annealing heuristic as given in Brusco and Jacobs (1993) and the branch-and-cut approach provided in Hoffman and Padberg (1993). For a more extensive overview of regular crew scheduling literature we refer to Ernst et al. (2004) and Van den Bergh et al. (2013).

3.2 Crew Rescheduling Problems

In case the underlying data changes or contains uncertainty a crew schedule can become infeasible. A real-world example is whenever disruptions occur or if crew suddenly takes a day off or is ill. In these cases the current schedule has to be rescheduled, since from a practical point of view it may be undesirable to construct a new schedule from scratch. An example of a rescheduling approach that incorporates robustness is presented in Ionescu and Kliewer (2011) and Dück et al. (2012) where the the increase in flexibility and stability of crew schedules is considered respectively.

To solve these type of problems meta-heuristics are considered in Moz and Pato (2007) and in Maenhout and Vanhoucke (2011). In Moz and Pato (2007) the feasibility of nurse rosters are recovered by an evolutionary meta-heuristic. In Maenhout and Vanhoucke (2011) a genetic algorithm is used wherein specific hard constraints are considered. The fitness score of the individuals is related to the similarity between the newly formed roster and the infeasible roster.

An exact approach is considered in Moz and Pato (2004) whenever at least one crew member informs that the assigned shifts cannot be performed. Their contribution to rescheduling literature is an integer programming model based on a multi-commodity flow formulation where the nodes of the network are aggregated. Lastly, a quasi-robust optimization algorithm is discussed in Veelenturf et al. (2014) where the authors study large scale real-time disruptions. They indicate that this problem is equivalent to a two-stage problem where in the first stage the different scenarios is considered and in the second the stage the true duration of the disruption is revealed. The quasi robustness comes forth from a prescribed number of rescheduled duties that are required to be recoverable.

3.3 Collaborative Traineeship Planning

Literature related to our problem consists of the collaborative traineeship planning aspect. An example of the collaborative aspect is given in Juang et al. (2007) where a genetic algorithm is used to arrange training programs for crew members. In Beliën and Demeulemeester (2004) and Beliën and Demeulemeester (2006) a heuristic branch-and-price strategy is used in combination with column generation to build long-term trainee schedules. At first sight their problem seems similar, however a closer analysis is that our problem has in addition varying internship duration, multiple locations per internship and related penalty, block structure on groups of internships and lastly the rescheduling aspect.

In Guo et al. (2014) a discussion of a basic residency problem is presented where a one-year schedule is produced for an $\mathcal{NP}$-complete problem. In case of shared human resources, Stolletz and Brunner (2012) consider a fair optimization of shifts over physicians. In this paper a comparison of a set covering approach is made with an implicit modelling technique, wherein shift building rules are implemented as constraints. Both techniques allow full flexibility in the terms of different starting times and lengths as well as break replacements. The scheduling model integrates physician preferences and fairness aspects with respect to crew.

4 Data Description

In this section we describe the data required for the CTPP. We start with the data required for an instance of the initial planning problem. Thereafter we elaborate on the additional data needed for an instance of the rescheduling problem.

For each instance of the CTPP an essential part consists of the time periods people are able to perform internships and which internships shall be done by these people. That is, we assume the following data is available for each person:

- Available time periods a person can perform internships,
- Type of internships a person shall perform in order to meet the program requirements,
- Range on the minimum and maximum number of time periods a person shall perform each internship type.

Next, we assume the following data is available for each internship type:

- The block an internship corresponds to,
- At which locations an internship type can be performed,
- Soft upper and lower staffing targets at each location on the number of people the location aims to have during each time period for each internship type.

As these staffing targets are considered soft targets, we require data related to the following penalties in order to identify importance of each aspect:

- Penalty for violating soft upper and lower internship staffing targets internship targets during each time period,
- Penalty for increased duration of someones traineeship program.

In case of disruptions such that we apply a rescheduling procedure we assume the following data is available:

- Change of availability per person,
- Change of soft upper- and lower staffing targets during a certain interval,
- Additional people in the training program for which the data as described for the initial planning problem is also known.

Lastly, we assume for the rescheduling case the corresponding penalty is set proportional to the penalties corresponding to initial planning problem.

5 Problem Formulation

In this section we describe the mathematical formulation of the problem. First, we introduce relevant notation in Section 5.1 − 5.4 to describe the sets, subsets, decision variables and parameters. Thereafter, the mathematical formulation corresponding to the initial planning problem is given in Section 5.5. This formulation is extended in Section 5.6 to incorporate the rescheduling component. Lastly, in Section 5.7 we prove both the initial planning as well as the rescheduling problem are $\mathcal{NP}$-complete.

5.1 Sets

In our mathematical formulation we use the following sets:

- $\mathcal{P}$: set of people,
- $\mathcal{I}$: set of internship types,
- $\mathcal{B}$: set of blocks.
- $\mathcal{L}$: set of locations,
- $\mathcal{T}$: set of time periods.

Let the size of each set be given as: $p_{max} = |\mathcal{P}|$, $i_{max} = |\mathcal{I}|$, $b_{max} = |\mathcal{B}|$, $l_{max} = |\mathcal{L}|$ and $t_{max} = |\mathcal{T}|$.

5.2 Subsets

In our mathematical formulation we use the following subsets:

- $\mathcal{P}_{b1}$: subset of people starting in the first block,
- $\mathcal{I}_p$: subset of internship types that person p has to perform,
- $\mathcal{I}_b$: subset of internship types that must be performed to complete block b,
- $\mathcal{I}_{b1}$: subset of internship types that must be performed to complete the first block,
- $\mathcal{B}_p$: subset of blocks that person p has to perform,
- $\mathcal{L}_i$: subset of locations where internship i can be performed,
- $\mathcal{T}_p$: subset of time periods person p is available.

5.3 Decision Variables

In our mathematical formulation we use the following decision variables:

$$x_{pilt} = \begin{cases} 1, & \text{if person } p \text{ performs internship type } i \text{ at location } l \text{ at time period } t, \\ 0, & \text{otherwise,} \end{cases}$$

$$y_{pit} = \begin{cases} 1, & \text{if person } p \text{ completed internship type } i \text{ by time period } t, \\ 0, & \text{otherwise,} \end{cases}$$

$$q_{pbt} = \begin{cases} 1, & \text{if person } p \text{ completed block } b \text{ by time period } t, \\ 0, & \text{otherwise,} \end{cases}$$

$$w_{pilt} = \begin{cases} 1, & \text{if person } p \text{ started internship type } i \text{ at location } l \text{ at time period } t, \\ 0, & \text{otherwise,} \end{cases}$$

$$f_{pt} = \begin{cases} 1, & \text{if person } p \text{ meets the program requirements by time period } t, \\ 0, & \text{otherwise,} \end{cases}$$

$$u^{+}_{ilt} = \text{number of over-staffed type } i \text{ internships at location } l \text{ at time period } t,$$

$$u^{-}_{ilt} = \text{number of under-staffed type } i \text{ internships at location } l \text{ at time period } t.$$

Note: decision variable x_{pilt} uniquely defines a solution to the CTPP, as it completely defines the assignment of people to internship types and locations over time. Hence, the other decision variables follow directly from a given realisation of x_{pilt}.

5.4 Parameters

In our mathematical formulation we use the following parameters:

- $\mathbf{MinP}_{pi}$: minimum number of time periods internship type i must be performed by person p,
- $\mathbf{MaxP}_{pi}$: maximum number of time periods internship type i can be performed by person p,
- $\mathbf{UB}_{ilt}$: soft upper target on the number of people to perform internship type i at location l during time period t,
- $\mathbf{LB}_{ilt}$: soft lower target on the number of people to perform internship type i at location l during time period t,
- $\mathbf{c}^{+}_{t}$: penalty per over-staffed internship during time period t,
- $\mathbf{c}^{-}_{t}$: penalty per under-staffed internship during time period t,

- $\mathbf{b}_p$: penalty for person p per time period late program finish,
- $\mathbf{e}_p$: maximum number of time periods person p can perform internships in the program.

These parameters represent the range on the number of periods an internship may be performed, the target on the number of internships available, the maximum number of violations on this target and the corresponding penalties.

5.5 Initial Planning Formulation

In this section we formulate the problem as a MIP. The objective function z of this problem is given by

$$z = \min \left\{ \sum_{i \in \mathcal{I}} \sum_{l \in \mathcal{L}_i} \sum_{t \in \mathcal{T}} (\mathbf{c}_t^+ \cdot u_{ilt}^+ + \mathbf{c}_t^- \cdot u_{ilt}^-) + \sum_{p \in \mathcal{P}} (\mathbf{b}_p \cdot (\mathbf{e}_p - \sum_{t \in \mathcal{T}_p} f_{pt})) \right\}. \tag{5.1}$$

The objective function is given by Equation (5.1) which is to minimize the total penalty for over- and understaffing internships plus the penalty for time periods a person completes the program later than the earliest possible time period that person can complete the program. Note that we ensure that $\sum_{p \in \mathcal{P}} (\mathbf{e}_p - \sum_{t \in \mathcal{T}_p} f_{pt}) \geq 0$, by restricting the number of available time periods per person. In the remainder of this section we will elaborate on Constraints (5.2) – (5.21), which define the feasible region.

$$f_{pt} + \sum_{i \in \mathcal{I}_p} \sum_{l \in \mathcal{L}_i} x_{pilt} = 1, \qquad \forall p \in \mathcal{P}, t \in \mathcal{T}_p, \tag{5.2}$$

$$\sum_{l \in \mathcal{L}_i} \sum_{t \in \mathcal{T}_p : t < t^*} x_{pilt} \geq \mathbf{MinP}_{pi} \cdot y_{pit^*}, \qquad \forall p \in \mathcal{P}, i \in \mathcal{I}_p, t^* \in \mathcal{T}_p, \tag{5.3}$$

$$\sum_{l \in \mathcal{L}_i} \sum_{t \in \mathcal{T}_p} x_{pilt} \leq \mathbf{MaxP}_{pi}, \qquad \forall p \in \mathcal{P}, i \in \mathcal{I}_p. \tag{5.4}$$

To ensure the consecutive training program we restrict in Constraints (5.2) that each person is busy with the training program during each time period, except for the time periods that person finished the training program. The minimum and maximum duration of each internship is fixed in Constraints (5.3) and (5.4). The following constraints will force the block structure on the internship order.

$$y_{pit} + \sum_{l \in \mathcal{L}_i} x_{pilt} \leq 1, \qquad \forall p \in \mathcal{P}, i \in \mathcal{I}_p, t \in \mathcal{T}_p, \tag{5.5}$$

$$y_{pit} \geq y_{pi(t-1)}, \qquad \forall p \in \mathcal{P}, i \in \mathcal{I}_p, t \in \mathcal{T}_p : (t-1) \in \mathcal{T}_p, \tag{5.6}$$

$$q_{pbt} \leq y_{pit}, \qquad \forall p \in \mathcal{P}, b \in \mathcal{B}_p, i \in \mathcal{I}_b, \mathcal{T}_p, \tag{5.7}$$

$$q_{pbt} \leq q_{p(b-1)t}, \qquad \forall p \in \mathcal{P}, b \in \mathcal{B}_p : (b-1) \in \mathcal{B}_p, t \in \mathcal{T}_p, \tag{5.8}$$

$$q_{pbt} \geq q_{pb(t-1)}, \qquad \forall p \in \mathcal{P}, b \in \mathcal{B}_p, t \in \mathcal{T}_p : (t-1) \in \mathcal{T}_p. \tag{5.9}$$

Constraints (5.5) ensures that if a person completed an internship by time t this person can only perform this internship until time t (excluding t itself). Constraints (5.6) ensures that once an internship is considered completed it remains completed in the next periods. Constraints (5.7) forces a block to be completed if a person completed all its corresponding internships of that block. Constraints (5.8) ensures that blocks are performed in the correct order. Constraints (5.9) ensures that a block remains completed in the next periods once it is considered as completed. We ensure each person to perform the internships of the training program in a feasible order, by restricting the variable indicating whether the training program is finished using the following constraints.

$$f_{pt} \leq q_{pbt}, \qquad \forall p \in \mathcal{P}, b \in \mathcal{B}_p, t \in \mathcal{T}_p, \tag{5.10}$$

$$\sum_{t \in \mathcal{T}_p} f_{pt} \geq 1, \qquad \forall p \in \mathcal{P}, \tag{5.11}$$

$$\sum_{i \in \mathcal{I}_b} \sum_{l \in \mathcal{L}_i} x_{pilt} \leq q_{p(b-1)t}, \qquad \forall p \in \mathcal{P}, b \in \mathcal{B}_p : (b-1) \in \mathcal{B}_p, t \in \mathcal{T}_p, \tag{5.12}$$

$$w_{pilt} \geq x_{pilt} - x_{pil(t-1)}, \qquad \forall p \in \mathcal{P}, i \in \mathcal{I}_p, l \in \mathcal{L}_i, t \in \mathcal{T}_p, \tag{5.13}$$

$$\sum_{l \in \mathcal{L}_i} \sum_{t \in \mathcal{T}_p} w_{pilt} \leq 1, \qquad \forall p \in \mathcal{P}, i \in \mathcal{I}_p. \tag{5.14}$$

Constraints (5.10) ensures that the program is considered as completed only if that person completed all its corresponding blocks of the training program. Constraints (5.11) make sure that each person completes the training program. Constraints (5.12) ensure that a person can only perform internships of its current block. Constraints (5.13) and (5.14) require each person to perform each internship consecutively. That is, a person can start an internship only once. This constraint also ensures that each person can perform an internship at only one location. Soft bounds on the number of internships available are imposed by the following constraints.

$$\sum_{p \in \mathcal{P}_i} x_{pilt} - \mathbf{UB}_{ilt} \leq u^+_{ilt}, \qquad \forall i \in \mathcal{I}, l \in \mathcal{L}_i, t \in \mathcal{T}, \tag{5.15}$$

$$\mathbf{LB}_{ilt} - \sum_{p \in \mathcal{P}} x_{pilt} \leq u^-_{ilt}, \qquad \forall i \in \mathcal{I}, l \in \mathcal{L}_i, t \in \mathcal{T}. \tag{5.16}$$

Constraints (5.15) and (5.16) ensure the decision variables corresponding to the number of internships over- and understaffed are set. Lastly, Constraints (5.17) − (5.21) below are the integrality constraints on the decision variables.

$$x_{pilt}, w_{pilt} \in \{0,1\}, \qquad \forall p \in \mathcal{P}, i \in \mathcal{I}_p, l \in \mathcal{L}_i, t \in \mathcal{T}_p, \tag{5.17}$$

$$y_{pit} \in \{0,1\}, \qquad \forall p \in \mathcal{P}, i \in \mathcal{I}_p, t \in \mathcal{T}_p, \tag{5.18}$$

$$q_{pbt} \in \{0,1\}, \qquad \forall p \in \mathcal{P}, b \in \mathcal{B}_p, t \in \mathcal{T}_p, \tag{5.19}$$

$$f_{pt} \in \{0,1\}, \qquad \forall p \in \mathcal{P}, t \in \mathcal{T}_p, \tag{5.20}$$

$$u^+_{ilt}, u^-_{ilt} \in \mathbb{N}_0, \qquad \forall i \in \mathcal{I}, l \in \mathcal{L}_i, t \in \mathcal{T}. \tag{5.21}$$

5.6 Rescheduling Formulation

To incorporate the rescheduling aspect in the mathematical formulation we introduce the parameter $\hat{x}_{pilt}$, which is 1 if in the original schedule person p performs an internship of type i at location l during time period t. Let $\hat{\mathbf{R}}_{pilt} = 1$, if person p is allowed to be rescheduled during time period t to perform internship i at location l, and 0 otherwise. Furthermore, let decision variable r_{pilt} be defined as

$$r_{pilt} = \begin{cases} 1, & \text{if we change person } p \text{ to perform internship } i \\ & \text{at location } l \text{ during time period } t, \\ 0, & \text{otherwise.} \end{cases}$$

The corresponding penalty $\mathbf{d}_{pilt}$ denotes the penalty incurred in case person p is rescheduled to internship i at location l during time period t from another internship or location at that time period. Thus, we add the following penalty term to the objective function given in (5.1),

$$\sum_{p \in \mathcal{P}} \sum_{i \in \mathcal{I}_p} \sum_{l \in \mathcal{L}_i} \sum_{t \in \mathcal{T}_p} \mathbf{d}_{pilt} \cdot r_{pilt}. \tag{5.22}$$

The penalty term given in (5.22) is to minimize the weighted number of rescheduled internships and locations. More specifically, each internship performed by each person at some location during a given time period is assigned a weight that indicates the penalty in case this person is rescheduled. That is, higher weights indicate that it is more undesirable to reschedule. In addition to Constraints (5.2) − (5.21) we impose the following extra constraints to the mathematical formulation:

$$r_{pilt} \geq x_{pilt} - \hat{x}_{pilt}, \qquad \forall p \in \mathcal{P}, i \in \mathcal{I}_p, l \in \mathcal{L}_i, t \in \mathcal{T}_p, \tag{5.23}$$

$$r_{pilt} \leq \hat{\mathbf{R}}_{pilt}, \qquad \forall p \in \mathcal{P}, i \in \mathcal{I}_p, l \in \mathcal{L}_i, t \in \mathcal{T}_p, \tag{5.24}$$

$$r_{pilt} \in \{0, 1\}, \qquad \forall p \in \mathcal{P}, i \in \mathcal{I}_p, l \in \mathcal{L}_i, t \in \mathcal{T}_p. \tag{5.25}$$

Constraints (5.23) ensure that if we reschedule person p during time period t to some other internship or location the penalty term $\mathbf{d}_{pilt}$ is activated in the objective function. Constraints (5.24) ensure that rescheduling is only permitted for designated decision variables. Note: Constraints (5.23) and (5.24) consider a generic rescheduling approach. For example, we could achieve that only location changes are permitted by setting $\hat{\mathbf{R}}_{pilt}$. The same intuition applies to the value of the penalty parameter $\mathbf{d}_{pilt}$. For example, if new people enter the program the penalty could be set to 0 for all new people as these were not in the original schedule. Lastly, Constraints (5.25) ensure the integrality of the rescheduling decision variables.

5.7 Complexity Analysis

In this section we elaborate on the complexity of initial planning problem as well as the rescheduling problem. We show that both problems are $\mathcal{NP}$-complete. First, we reduce 2-PARTITION to the initial planning problem. Thereafter, we show that the initial planning problem can easily be reduced to the rescheduling problem.

5.7.1 Partition Problem

To show that the initial planning problem is $\mathcal{NP}$-complete we use the well-known $\mathcal{NP}$-complete 2-PARTITION problem (see Karp (1972)). The 2-PARTITION decision problem is defined below.

Definition 5.1. The 2-PARTITION decision problem is: given a set $\mathcal{S}$ of integers $a_1, \ldots, a_m$ and an integer A such that $\sum_{i \in \mathcal{S}} a_i = 2A$ and $a_i \leq A, \forall i \in \mathcal{S}$. Is there a partition $\mathcal{S}_1, \mathcal{S}_2$ such that $\mathcal{S}_1 \cup \mathcal{S}_2 = \mathcal{S}$, $\mathcal{S}_1 \cap \mathcal{S}_2 = \emptyset$ and $\sum_{i \in \mathcal{S}_1} a_i = \sum_{j \in \mathcal{S}_2} a_j = A$?

5.7.2 Initial Planning Problem

Let the decision problem of the initial planning problem be defined as: is there a solution for the initial planning problem for which objective function (see Equation (5.1)) is 0?

Theorem 5.2. *The initial planning problem is $\mathcal{NP}$-complete.*

Proof. We show the initial planning problem is $\mathcal{NP}$-complete as follows:

(i) First, we show the decision problem is in $\mathcal{NP}$.

(ii) Second, we show the 2-PARTITION problem can be reduced in polynomial time to an instance of the initial planning problem.

(i) We prove that the decision problem is in $\mathcal{NP}$, by showing that the certificate to the decision problem can be verified in polynomial time. First, note that each solution of the initial planning problem can be uniquely represented by the decision variable x_{pilt} as described in Section 5.3 (recall: that $x_{pilt} = 1$, if person p performs internship i at location l during time period t). The certificate can be verified using this representation by checking if the objective function equals 0 (resulting in 'Yes' as answer) or is greater than 0 (resulting in 'No' as answer). Rewriting the objective function as given in Equation (5.1) gives:

$$\begin{aligned}
&\sum_{i\in\mathcal{I}}\sum_{l\in\mathcal{L}_i}\sum_{t\in\mathcal{T}}\mathbf{c}_t^{+}\cdot\max\left\{0,\ \sum_{p\in\mathcal{P}}x_{pilt}-\mathbf{UB}_{ilt}\right\}\\
+&\sum_{i\in\mathcal{I}}\sum_{l\in\mathcal{L}_i}\sum_{t\in\mathcal{T}}\mathbf{c}_t^{-}\cdot\max\left\{0,\ \mathbf{LB}_{ilt}-\sum_{p\in\mathcal{P}}x_{pilt}\right\}\\
+&\sum_{p\in\mathcal{P}}\mathbf{b}_p\cdot\left(\mathbf{e}_p-\sum_{t\in\mathcal{T}_p}\min\left\{1,\ 1-\sum_{i\in\mathcal{I}_p}\sum_{l\in\mathcal{L}_i}x_{pilt}\right\}\right).
\end{aligned}\tag{5.26}$$

Note that (5.26) can be calculated in polynomial time and that therefore the certificate can be verified in polynomial time. Hence, the decision problem is in $\mathcal{NP}$.

(ii) We show that the 2-PARTITION problem can be reduced in polynomial time to an instance of the initial planning problem. That is, we proof that the answer to the 2-PARTITION problem is 'Yes' $\iff$ the answer to the initial planning problem is 'Yes'.

Consider the following instance of the initial planning problem with 2 internships, 2 locations, 1 block and m people, thus $\mathcal{P} = \{1, \ldots, m\}$. Let each person be available during any time period. That is, $\mathcal{T}_p = \mathcal{T}, \forall p \in \mathcal{P}$. Now, let the durations of internships $\forall p \in \mathcal{P}$ be restricted as:

- $\mathbf{MinP}_{p1} = \mathbf{MaxP}_{p1} = a_p$, for some integer a_p such that $\sum_{p^*\in P} a_{p^*} = 2A$ for some integer A,
- $\mathbf{MinP}_{p2} = 1$,
- $\mathbf{MaxP}_{p2} = 2A$.

This structure enables each person to start internship 1 at any time period $t \in \{1, \ldots, 2A\}$. Let the penalty associated per time period of late program finish be $\mathbf{b}_p = 0, \forall p \in \mathcal{P}$. Let the penalty for violating staffing targets be given as $\mathbf{c}_t^{+} =$

$\mathbf{c}_t^- = 1$, $\forall t \in \mathcal{T}$. Assume both internships can be done at each of the two locations, where the upper and lower target $\mathbf{UB}_{ilt}$ and $\mathbf{LB}_{ilt}$ (also referred to as capacity) for internship i at location l during time period t are as follows:

- Let $\mathbf{LB}_{2lt} = 0$, $\forall l \in \mathcal{L}, t \in \mathcal{T}$.
- Let $\mathbf{UB}_{2lt} = m$, $\forall l \in \mathcal{L}, t \in \mathcal{T}$.
- Let $\mathbf{LB}_{11t} = \mathbf{UB}_{11t} = 1$, $\forall t \in \{1, \dots, A\}$ and 0 otherwise.
- Let $\mathbf{LB}_{12t} = \mathbf{UB}_{12t} = 1$, $\forall t \in \{A+1, \dots, 2A\}$ and 0 otherwise.

Figure 1 below illustrates the capacity for internship 1.

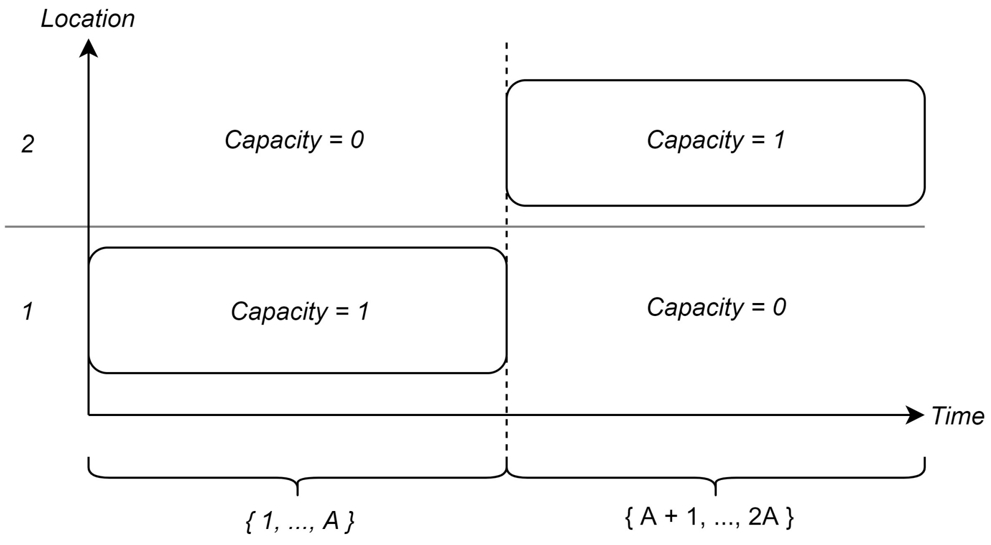


Figure 1: Illustration of internship 1 capacity at location 1 and 2

$\Longrightarrow$ First, we show that if the answer to the initial planning problem is 'Yes' then the answer to the 2-PARTITION problem is 'Yes'. Assume the answer to the initial planning problem is 'Yes', with a given representation of the solution by the decision variable x_{pilt}, then the solution to the 2-PARTITION problem is found by partitioning $\mathcal{S}$ as:

$$\mathcal{S}_1 = \left\{ a_p \;\middle|\; \sum_{t \in \{1,\dots,A\}} x_{p11t} = a_p \right\}, \tag{5.27}$$

$$\mathcal{S}_2 = \left\{ a_p \;\middle|\; \sum_{t \in \{A+1,\dots,2A\}} x_{p12t} = a_p \right\}. \tag{5.28}$$

Note that $\mathcal{S}_1 \cup \mathcal{S}_2 = \mathcal{S}$ and $\mathcal{S}_1 \cap \mathcal{S}_2 = \emptyset$ since internship 1 must be performed at exactly one of the two locations. Since the answer to the initial planning problem is 'Yes' the total penalty is 0. Hence, $\max\left\{0,\ \sum_{p \in P} x_{pilt} - \mathbf{UB}_{ilt}\right\} = 0$, $\forall i \in$

$\mathcal{I}, l \in \mathcal{L}_i, t \in \mathcal{T}$ and $\max\left\{0,\ \mathbf{LB}_{ilt} - \sum_{p\in P} x_{pilt}\right\} = 0,\ \forall i \in \mathcal{I}, l \in \mathcal{L}_i, t \in \mathcal{T}$. Furthermore, using $\mathbf{LB}_{11t} = \mathbf{UB}_{11t} = 1, \forall t \in \{1,\ldots,A\}$ and $\mathbf{LB}_{12t} = \mathbf{UB}_{12t} = 1, \forall t \in \{A+1,\ldots,2A\}$ it follows that:

$$\sum_{p\in\mathcal{P}} x_{p11t} = 1, \qquad \forall t \in \{1,\ldots,A\}, \tag{5.29}$$

$$\sum_{p\in\mathcal{P}} x_{p11t} = 0, \qquad \forall t \in \{A+1,\ldots,2A\}, \tag{5.30}$$

$$\sum_{p\in\mathcal{P}} x_{p12t} = 1, \qquad \forall t \in \{A+1,\ldots,2A\}, \tag{5.31}$$

$$\sum_{p\in\mathcal{P}} x_{p12t} = 0, \qquad \forall t \in \{1,\ldots,A\}. \tag{5.32}$$

Combining Equations (5.27) − (5.32) it follows that:

$$\sum_{i\in\mathcal{S}_1} a_i = \sum_{p\in P} \sum_{t\in\{1,\ldots,A\}} x_{p11t} = \sum_{t\in\{1,\ldots,A\}} 1 = A, \tag{5.33}$$

$$\sum_{j\in\mathcal{S}_2} a_j = \sum_{p\in P} \sum_{t\in\{A+1,\ldots,2A\}} x_{p12t} = \sum_{t\in\{A+1,\ldots,2A\}} 1 = A. \tag{5.34}$$

The reduction is graphically illustrated for $\mathcal{P} = \{1,\ldots,7\}$ in Figure 2 below, where person 1, 5 and 7 are assigned to location 1 (and the corresponding integers of the 2-PARTITION problem to $\mathcal{S}_1$) and person 2, 3, 4 and 6 are assigned to location 2 (and the corresponding integers of the 2-PARTITION problem to $\mathcal{S}_2$).

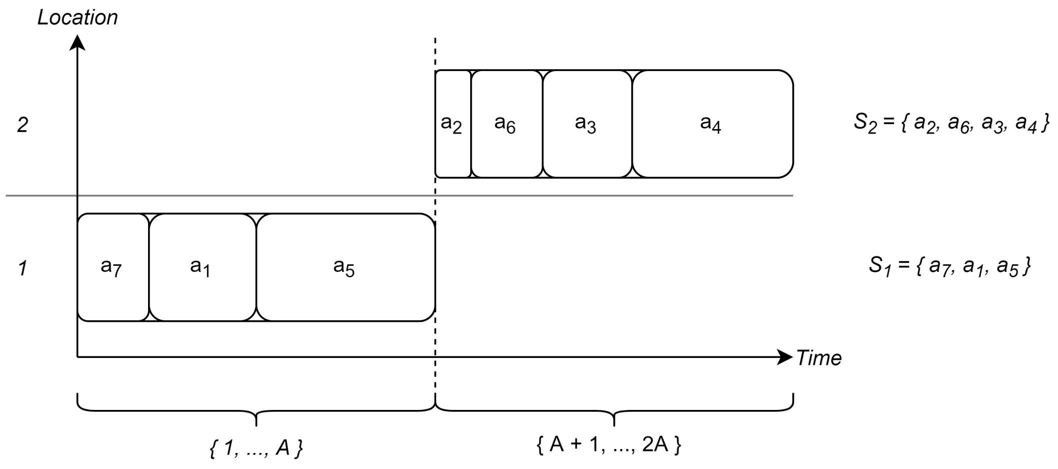


Figure 2: Intuition of the polynomial reduction

$\Longleftarrow$ Second, we show that if the answer to the 2-PARTITION problem is 'Yes', then the answer to the initial planning problem instance is 'Yes'. Assume the answer to the 2-PARTITION problem is 'Yes'. Again we use the intuition as illustrated in

Figure 2. The solution to the initial planning problem can be constructed as follows: Assign person p to location 1 if $a_p \in \mathcal{S}_1$ and to location 2 if $a_p \in \mathcal{S}_2$. Furthermore, by appropriately choosing the order and duration of internship 2 we can assure that person p can start internship 1 during any time period $t^* \in \mathcal{T}_p$ at the assigned location, as illustrated in Figure 3 below.

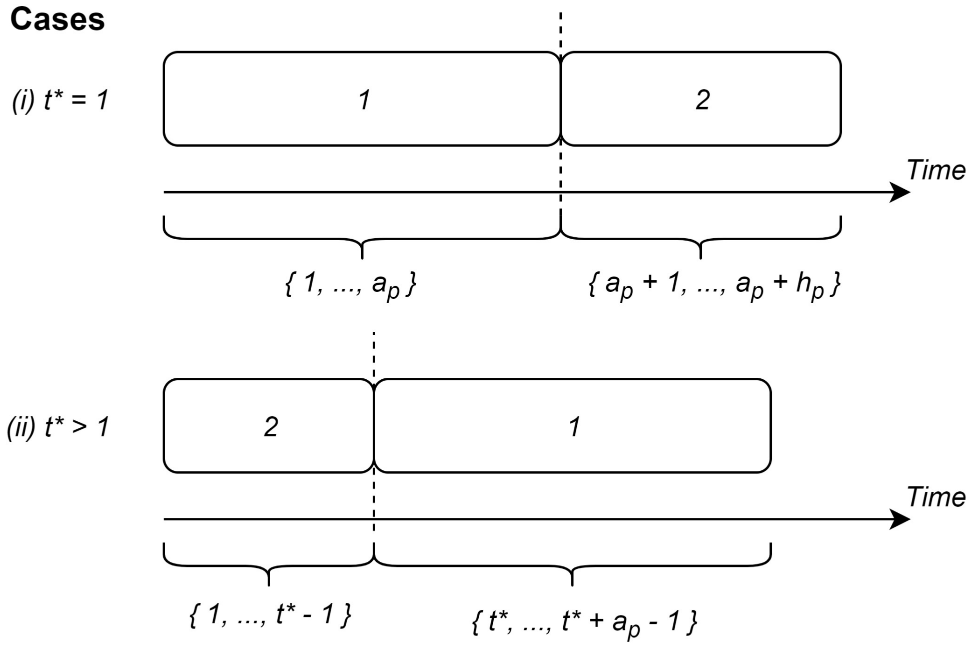


Figure 3: Illustration of both cases for starting internship 1 at time period t^*

That is, if person p should start internship 1 at *(i)* $t^* = 1$, then the order should be $(1, 2)$ and the duration of internship 2 can be arbitrarily chosen as h_p. If person p should start internship 1 at time period *(ii)* $t^* > 1$, then the order should be $(2, 1)$ and the duration of internship 2 is set to t^* such that person p starts internship 1 at time period t^*. Note that internship 2 can be assigned to either of the 2 locations since both targets $\mathbf{LB}_{2lt}$ and $\mathbf{UB}_{2lt}$ will not be violated. Using this result and the fact that $\sum_{i \in \mathcal{S}_1} a_i = \sum_{j \in \mathcal{S}_2} a_j = A$ an order of people can be created such that equations (5.29) – (5.32) hold (for example by sorting for each location the people assigned to that location in increasing order of internship duration and let person p_2 perform internship 1 directly after person p_1 finished internship 1). Combining Equations (5.29) – (5.32) and $\mathbf{LB}_{11t} = \mathbf{UB}_{11t} = 1, \forall t \in \{1, \ldots, A\}$ and $\mathbf{LB}_{12t} = \mathbf{UB}_{12t} = 1, \forall t \in \{A+1, \ldots, 2A\}$ results in $\max\left\{0,\ \sum_{p \in P} x_{pilt} - \mathbf{UB}_{ilt}\right\} = 0$ and $\max\left\{0,\ \mathbf{LB}_{ilt} - \sum_{p \in P} x_{pilt}\right\} = 0$. Thus, constructing the initial planning problem instance this way results in a total penalty of 0. Hence, we have proven that the 2-PARTITION problem can be reduced in polynomial time to an instance of the initial planning problem. $\square$

5.7.3 Rescheduling Problem

Theorem 5.3. *The rescheduling problem is $\mathcal{NP}$-complete*

Proof. Let the decision problem be: is there a solution with penalty at most k? First, note that the rescheduling problem is in $\mathcal{NP}$ since a certificate can be verified in polynomial time by calculating the objective function as given in Equation (5.26) plus the the rescheduling term given in Equation (5.22) which can be computed in polynomial time. Second, the initial planning problem can be polynomially reduced to the following instance of the rescheduling problem. Let the penalty associated to each rescheduling change be 0 (thus $d_{pilt} = 0$ in Section 5.6). Allow that all changes of the initial schedule are possible (by setting $\hat{\mathbf{R}}_{pilt} = 1$). For this instance of the rescheduling problem it holds that this is exactly equal to the initial planning problem. Using Theorem 5.2 it can be easily verified that the rescheduling problem is also $\mathcal{NP}$-complete. $\square$

6 Methodology

In this section we describe the methodology to solve the problem. First, we consider the initial planning problem in Section 6.1. Next, we elaborate on the rescheduling problem in Section 6.2.

6.1 Initial Planning Problem

In this section we describe the methodology for the initial planning problem. First, we elaborate on an exact approach in Section 6.1.1, wherein we use the mathematical formulation as described in Section 5.5. To improve on the performance of this approach, we take advantage of the problem structure by fixing some decision variables and introduce two valid inequalities.

Next, the $\mathcal{NP}$-completeness proof in Section 5.7.2 gives rise to explore a heuristic procedure which we discuss in Section 6.1.2. In this procedure we use a genetic algorithm followed by a mathematical programming heuristic to solve the initial planning problem. In the genetic algorithm we decompose the problem into three stages, where the first two stages correspond to a master-subproblem structure and the last stage to an improvement strategy. In the master problem we decide for each person on the order in which internships are performed and the corresponding duration. The remaining subproblem corresponds to the assignment of each internship of each person to a location for a fixed order and duration of each internship. After the subproblem is solved we utilize a local search strategy on the internship duration per person for each schedule. Lastly, we improve on the best schedule resulting from this procedure by exploiting a mathematical programming based heuristic.

6.1.1 Exact Approach

In this section we outline the exact approach to solve the initial planning problem. The MIP formulation of the problem as presented in Section 5.5 can be solved by applying a branch-and-bound algorithm. However, as the number of decision variables can be high and the problem is $\mathcal{NP}$-complete (see Section 5.7.2), we intend to reduce computation time by exploiting variable fixing methods in Section 6.1.1.1. Second, we introduce two valid inequalities in Section 6.1.1.2 to tighten the mathematical formulation.

6.1.1.1 Variable Fixing

In this section we describe the variable fixing methods we exploit to improve the performance of the branch-and-bound algorithm. For each of the variable fixing methods we take advantage of the minimum and maximum duration of each internship per person. The following constraints show how decision variable f_{pt}, indicating if person p finished the program by time period t, is fixed.

$$f_{pt} = 0, \qquad \text{if} \sum_{i \in \mathcal{I}_p} \mathbf{MinP}_{pi} \leq \sum_{t' \in \mathcal{T}_p : t' \leq t} 1, \qquad \forall p \in \mathcal{P}, t \in \mathcal{T}_p, \tag{6.1}$$

$$f_{pt} = 1, \qquad \text{if} \sum_{i \in \mathcal{I}_p} \mathbf{MaxP}_{pi} > \sum_{t' \in \mathcal{T}_p : t' \leq t} 1, \qquad \forall p \in \mathcal{P}, t \in \mathcal{T}_p. \tag{6.2}$$

First, in Constraints (6.1) we use that person p cannot finish the program by time period t, if the number of time periods that is at least required to finish the program is less than or equal to the number of time periods person p is available up-to and including time period t. The number of time periods that is at least required is sum of the minimum internship durations for person p and is given by the left-hand side of Constraints (6.1). The number of time periods person p is available up-to and including time period t is given by the right-hand side of Constraints (6.1) and will be used in each of the variable fixing methods in this section. Second, in Constraints (6.2) we exploit the opposite effect of Constraints (6.1). Herein we use that person p must have finished the program by time period t if the number of time periods that is at most required to finish the program is more than the number of time periods person p is available up-to and including time period t.

The next decision variable on which variable fixing is applied denotes if person p finished block b by time period t. The corresponding constraints are given below.

$$q_{pbt} = 0, \quad \text{if} \sum_{b' \in \mathcal{B}_p : b' \leq b} \; \sum_{i \in \mathcal{I}_p \cap \mathcal{I}_{b'}} \mathbf{MinP}_{pi} \leq \sum_{t' \in \mathcal{T}_p : t' \leq t} 1, \quad \forall p \in \mathcal{P}, b \in \mathcal{B}_p, t \in \mathcal{T}_p, \tag{6.3}$$

$$q_{pbt} = 1, \quad \text{if} \sum_{b' \in \mathcal{B}_p : b' \leq b} \; \sum_{i \in \mathcal{I}_p \cap \mathcal{I}_{b'}} \mathbf{MaxP}_{pi} > \sum_{t' \in \mathcal{T}_p : t' \leq t} 1, \quad \forall p \in \mathcal{P}, b \in \mathcal{B}_p, t \in \mathcal{T}_p. \tag{6.4}$$

In contrast to Constraints (6.1) and (6.2), we also take advantage of the block-wise structure of the problem in Constraints (6.3) and (6.4). That is, in Constraints (6.3) we use that person p cannot finish block b by time period t if the sum of the minimum durations of the internships in blocks $b' \in \mathcal{B} : b' \leq b$ are less than or equal to the number of available time periods up-to and including time period t. A similar argument is applied in Constraints (6.4).

For the next variable fixing method we first introduce some notation. Let $b_i \in \mathcal{B}$ denote the block corresponding to internship i. Furthermore, let $\mathcal{B}_{pi} \subseteq \mathcal{B}$ and $\mathcal{B}^+_{pi} \subseteq \mathcal{B}$

be defined as

$$\mathcal{B}_{pi} = \left\{ b \in \mathcal{B} \,\middle|\, b < b_i,\, b \in \mathcal{B}_p \right\}, \qquad \forall p \in \mathcal{P}, i \in \mathcal{I}_p, \tag{6.5}$$

$$\mathcal{B}^+_{pi} = \mathcal{B}_{pi} \cup \{b_i\}, \qquad \forall p \in \mathcal{P}, i \in \mathcal{I}_p. \tag{6.6}$$

These definitions are illustrated in Example 6.1 below.

Example 6.1. Let $\mathcal{B} = \mathcal{B}_p = \{1, \dots, 3\}$. Furthermore, let $\mathcal{I} = \mathcal{I}_p = \{1, \dots, 6\}$ such that b_i is as follows $b_i = \left\lceil \frac{i}{2} \right\rceil$. Then, b_i, $\mathcal{B}_{pi}$ and $\mathcal{B}^+_{pi}$ are defined as:

i	1	2	3	4	5	6
b_i	1	1	2	2	3	3
$\mathcal{B}_{pi}$	$\emptyset$	$\emptyset$	{1}	{1}	{1, 2}	{1, 2}
$\mathcal{B}^+_{pi}$	{1}	{1}	{1, 2}	{1, 2}	{1, 2, 3}	{1, 2, 3}

Using $\mathcal{B}_{pi}$ and $\mathcal{B}^+_{pi}$, we fix the decision variable y_{pit} (recall that this variable indicates whether person p finished internship i by time period t) as follows.

$$y_{pit} = 0, \text{ if } \mathbf{MinP}_{pi} + \sum_{b \in \mathcal{B}_{pi}} \sum_{i' \in \mathcal{I}_p \cap \mathcal{I}_b} \mathbf{MinP}_{pi'} \leq \sum_{t' \in \mathcal{T}_p : t' \leq t} 1, \quad \forall p \in \mathcal{P}, i \in \mathcal{I}_p, t \in \mathcal{T}_p, \tag{6.7}$$

$$y_{pit} = 1, \text{ if } \sum_{b \in \mathcal{B}^+_{pi}} \sum_{i' \in \mathcal{I}_p \cap \mathcal{I}_b} \mathbf{MaxP}_{pi'} > \sum_{t' \in \mathcal{T}_p : t' \leq t} 1, \quad \forall p \in \mathcal{P}, i \in \mathcal{I}_p, t \in \mathcal{T}_p. \tag{6.8}$$

Again, we take advantage of the duration of each internship and block structure. Note the similarity between Constraints (6.3) – (6.4) and Constraints (6.7) – (6.8). However, in contrast to Constraints (6.3) we have to correct for the unknown order of internships in Constraints (6.7). That is, in Constraints (6.7) we ensure that person p cannot finish internship i by time period t if the sum of the minimum duration of the internships in the blocks preceding internship i (blocks $\mathcal{B}_{pi}$) plus the minimum duration of internship i is less than the number of time periods person p is available up-to and including time period t.

Lastly, we fix the decision variables indicating whether person p performs internship i at location l during time period t. The variable fixing method for this variable is as follows.

$$x_{pilt} = 0, \text{ if } \sum_{b \in \mathcal{B}_{pi}} \sum_{i' \in \mathcal{I}_p \cap \mathcal{I}_b} \mathbf{MinP}_{pi'} \leq \sum_{t' \in \mathcal{T}_p : t' \leq t} 1, \quad \forall p \in \mathcal{P}, i \in \mathcal{I}_p, l \in \mathcal{L}_i, t \in \mathcal{T}_p, \tag{6.9}$$

$$x_{pilt} = 0, \text{ if } \sum_{b \in \mathcal{B}^+_{pi}} \sum_{i' \in \mathcal{I}_p \cap \mathcal{I}_b} \mathbf{MaxP}_{pi'} > \sum_{t' \in \mathcal{T}_p : t' \leq t} 1, \quad \forall p \in \mathcal{P}, i \in \mathcal{I}_p, l \in \mathcal{L}_i, t \in \mathcal{T}_p. \tag{6.10}$$

Observe that we again exploit the minimum and maximum duration of each internship. In Constraints (6.9) we ensure that person p is not performing internship i by time period t if the internships in the blocks preceding (denoted by $\mathcal{B}_{pi}$) the block of internship i (denoted by b_i) cannot be finished by time period t. A similar argument holds for Constraints (6.10), wherein also the durations of the internships in block b_i itself are incorporated. Combining Constraints (5.13), (6.9) and (6.10) we note that the decision variable w_{pilt} (recall that w_{pilt} denotes if person p start internship i at location l at time period t) is fixed as well.

6.1.1.2 Valid Inequalities

In this section we describe the valid inequalities used to improve the performance of the branch-and-bound algorithm. First, let $\delta(p, t', t)$ be the number of time periods person p is available from time period t' up to and including time period t. Furthermore, let the sets $\mathcal{T}'_{pit} \subseteq \mathcal{T}_p$ and $\mathcal{T}''_{pit} \subseteq \mathcal{T}_p$ be defined as

$$\mathcal{T}'_{pit} = \left\{ t' \in \mathcal{T}_p \;\middle|\; t' \leq t,\ \delta(p, t', t) \leq \mathbf{MinP}_{pi} \right\}, \quad \forall p \in \mathcal{P}, i \in \mathcal{I}_p, t \in \mathcal{T}_p, \tag{6.11}$$

$$\mathcal{T}''_{pit} = \left\{ t'' \in \mathcal{T}_p \;\middle|\; t'' \leq t,\ \delta(p, t'', t) \leq \mathbf{MaxP}_{pi} \right\}, \quad \forall p \in \mathcal{P}, i \in \mathcal{I}_p, t \in \mathcal{T}_p. \tag{6.12}$$

We illustrate these sets in Example 6.2 below.

Example 6.2. Let the set of all time periods be $\mathcal{T} = \{1, \ldots, 5\}$. Furthermore, assume the availability of person p is $\mathcal{T}_p = \{1, 3, 4, 5\}$ and that the minimum and maximum duration internship i are $\mathbf{MinP}_{pi} = 2$ and $\mathbf{MaxP}_{pi} = 3$, respectively. Then the sets $\mathcal{T}'_{pit}$ and $\mathcal{T}''_{pit}$ are defined as:

t	1	2	3	4	5
$\mathcal{T}'_{pit}$	$\{1\}$	N.A.	$\{1, 3\}$	$\{3, 4\}$	$\{4, 5\}$
$\mathcal{T}''_{pit}$	$\{1\}$	N.A.	$\{1, 3\}$	$\{1, 3, 4\}$	$\{3, 4, 5\}$

The first valid inequality is then given by

$$x_{pilt} \geq \sum_{t' \in \mathcal{T}'_{pit}} w_{pilt'} \qquad \forall p \in \mathcal{P}, i \in \mathcal{I}_p, l \in \mathcal{L}, t \in \mathcal{T}_p. \tag{6.13}$$

The intuition of this inequality is as follows. Observe that $\mathcal{T}'_{pit}$ is the set of time periods from $t' = t$ back to $t' \in \mathcal{T}_p$ such that the number of time periods person p is available between t' and t is at most $\mathbf{MinP}_{pi}$. Then if person p starts internship i during some time period $t^* \in \mathcal{T}'_{pit}$, then person p should be performing internship i at time period t. We illustrate this valid inequality in Example 6.3 below.

Example 6.3. Assume person p is available during any time period and starts internship i during time period t^* at location l. That is, $w_{pilt^*} = 1$. Let $\mathbf{MinP}_{pi} = 3$. Then Constraints (6.13) imply the following for decision variable x_{pilt}:

t	$t^* - 1$	t^*	$t^* + 1$	$t^* + 2$	$t^* + 3$	$t^* + 4$	$t^* + 5$
w_{pilt}		1					
x_{pilt}		≥ 1	≥ 1	≥ 1			

The second valid inequality exploits the complementary effect of the first valid inequality and is given by

$$x_{pilt} \leq \sum_{t'' \in \mathcal{T}''_{pit}} w_{pilt''} \qquad \forall p \in \mathcal{P}, i \in \mathcal{I}_p, l \in \mathcal{L}, t \in \mathcal{T}_p. \tag{6.14}$$

Observe that $\mathcal{T}''_{pit}$ is the set of time periods from $t'' = t$ back to $t'' \in \mathcal{T}_p$ such that the number of time periods person p is available between t'' and t is at most $\mathbf{MaxP}_{pi}$. Then if person p does not start internship i during any time period $t^* \in \mathcal{T}''_{pit}$, then person p cannot be performing should be performing internship i at time period t. We illustrate this valid inequality in Example 6.4 below.

Example 6.4. Assume person p is available during any time period and starts internship i during time period t^* at location l. That is, $w_{pilt^*} = 1$. Let $\mathbf{MaxP}_{pi} = 4$. Then Constraints (6.12) imply the following for decision variable x_{pilt}:

t	$t^* - 1$	t^*	$t^* + 1$	$t^* + 2$	$t^* + 3$	$t^* + 4$	$t^* + 5$
w_{pilt}		1					
x_{pilt}	≤ 0					≤ 0	≤ 0

6.1.2 Heuristic Approach

In this section we elaborate on the heuristic approach to solve the initial planning problem. First, we discuss the intuition of the algorithm in Section 6.1.2.1 and introduce some relevant notation. Thereafter, we describe an overview of the algorithm in Section 6.1.2.2. The components of this algorithm are discussed in Sections 6.1.2.3 – 6.1.2.8.

6.1.2.1 Intuition of the Algorithm

In this section we describe the intuition of the heuristic approach and introduce some notation. First, observe that each solution to the initial planning problem can be uniquely represented by the joint outcome of the following three decisions:

- Order of internships per person,
- Duration of each internship per person,
- Location assignment of each internship per person.

To deal with the complexity of these decisions we decompose the problem into a master-subproblem structure. Next, we note the location assignment problem is a relatively easy problem once the order and duration of internships is given. Hence, we propose to decide on the order and duration of internships in the master problem and thereafter solve the location assignment problem as a subproblem. However, simultaneously deciding on the order and duration of internships is complicated as the quality of this can only be evaluated after the location subproblem is solved. That is, the over- and understaffing penalties are unknown before the location subproblem is solved. Therefore we apply a genetic algorithm to simultaneously decide on the order and duration of internships where we emphasize on the order of internships in the cross-over operation and use an extra improvement stage after the location subproblem is solved to optimize the duration of internships. Lastly, we end our heuristic approach by applying a mathematical programming heuristic to the best schedule resulting from the genetic algorithm to further improve on the duration of internships.

In summary, we decide on the order of internships in the first stage of the genetic algorithm. In the second stage we solve the location subproblem for the newly constructed schedule. Next, we improve on the internship durations of the newly formed schedule in the third stage of the algorithm. Lastly, the best schedule resulting from a certain number of iterations of the algorithm is improved by a mathematical programming heuristic. Note: during experimentation we observed it is favourable to apply the mathematical programming heuristic (for a reasonable amount of solution time) to a single schedule rather than applying the heuristic to multiple schedules (with distributed solution time).

In the remainder of this section we use the following

- θ : number of genetic algorithm iterations,
- κ : number of schedules in initial population,
- $\mathcal{S}$: set of schedules,
- $z(s)$ objective value of schedule s,
- v_{pil} : variable indicating whether person p performs internship i at location l,

- $\mathbf{v}_{pil}$: **parameter** indicating whether person p performs internship i at location l,
- g_{pit} : variable indicating whether person p performs internship i during time period t,
- $\mathbf{g}_{pit}$: **parameter** indicating whether person p performs internship i during time period t,
- ξ_{ilt} : temporary upper target on the number of internships of type i at location l during time period t,
- ζ_{ilt} : temporary lower target on the number of internships of type i at location l during time period t,
- $\mathcal{ED}_{pi}$: set of evaluated internship durations for person p internship type i in internship duration heuristic,
- $\mathcal{O}_p$: set of ordered pairs of internships in mathematical programming heuristic.

6.1.2.2 Genetic Algorithm

In this section we describe the overview of the three-stage genetic algorithm. The genetic algorithm is initialized by constructing a population of schedules as described in Section 6.1.2.3. Thereafter, in each iteration of the genetic algorithm $\kappa/2$ new schedules are formed. Each new schedule is formed by taking two schedules of the population and combine these into a new schedule as described in Section 6.1.2.4. We ensure in each iteration of the genetic algorithm each schedule is used exactly once in the cross-over operation. To each of the newly formed schedules mutation on the order and duration of internships per person is applied. Thereafter, the location subproblem is solved. This problem is solved using a greedy heuristic in case $iterations < \theta$ (see Section 6.1.2.5) or by an exact approach in case $iterations = \theta$ (see Section 6.1.2.6). The internship durations of each schedule are then improved by exploiting a local search strategy as described in Section 6.1.2.7. Next, the algorithm removes $\kappa/2$ schedules with the least objective values at the end of each iteration. Lastly, the best schedule resulting from the described procedure is improved by means of a mathematical programming based heuristic as described in Section 6.1.2.8. An overview of the heuristic approach is given in Algorithm 1.

Algorithm 1: Genetic Algorithm

```
Input: Genetic iterations ($\theta$), initial population size ($\kappa$), allowed internship
  durations per person ($\{\mathbf{Min}_{pi}, \mathbf{Max}_{pi}\}$), over- and understaffing penalties
  ($\mathbf{c}_t^+, \mathbf{c}_t^-$), internship lower- and upper target ($\mathbf{LB}_{ilt}, \mathbf{UB}_{ilt}$) per location.
Output: Best schedule $s_{best}$
$\mathcal{S} \leftarrow \{s_1, \ldots, s_\kappa\}$
Construct initial schedules
$\mathcal{S}' \leftarrow \mathcal{S}$
while $iterations \leq \theta$ do
    while $|\mathcal{S}| \geq 2$ do
        Select random schedules $s_1, s_2 \in \mathcal{S} : s_1 \neq s_2$
        Combine schedules $s_1$ and $s_2$ into $s_{new}$
        Mutate order and duration of internships of schedule $s_{new}$
        Solve location problem for schedule $s_{new}$
        $\mathcal{S}' \leftarrow \mathcal{S}' \cup s_{new}$
        $\mathcal{S} \leftarrow \mathcal{S} \setminus \{s_1, s_2\}$
    end
    for $s \in \mathcal{S}'$ do
        Duration improvement heuristic for schedule $s$
    end
    while $|\mathcal{S}'| \geq \kappa$ do
        $s_{delete} \leftarrow \arg\max_{s \in \mathcal{S}'} \Big\{ z(s) \Big\}$
        $\mathcal{S}' \leftarrow \mathcal{S}' \setminus s_{delete}$
    end
    $\mathcal{S} \leftarrow \mathcal{S}'$
end
$s_{best} \leftarrow \arg\min_{s \in \mathcal{S}} \Big\{ z(s) \Big\}$
Apply mathematical programming heuristic on schedule $s_{best}$
```

6.1.2.3 Construct Initial Schedules

In this section we describe the algorithm used to construct the initial population. For each schedule we generate for each person a random order of internships satisfying the block-wise structure property. For each of these internships a random duration is drawn within the range for that person. With the order and duration of each internship, the corresponding time periods person p performs internship i are derived, indicated by $\mathbf{g}_{pit}$. This parameter serves as input to the location subproblem. The construction algorithm is given in Algorithm 2 below.

Algorithm 2: Construct Initial Schedules

```
Input: Internships per person and range of duration ($\{\mathbf{MinP}_{pi}, \mathbf{MaxP}_{pi}\}$)
Output: Set of initial constructed schedules $\mathcal{S}$
for $s \in \mathcal{S}$ do
    for $p \in \mathcal{P}$ do
        for $b \in \mathcal{B}_p$ do
            $\tilde{\mathcal{I}} \leftarrow \mathcal{I}_p \cap \mathcal{I}_b$
            while $|\tilde{\mathcal{I}}| \geq 1$ do
                Draw random internship $i \in \tilde{\mathcal{I}}$
                Draw random internship duration $rd$ in $\left\{\mathbf{MinP}_{pi}, \mathbf{MaxP}_{pi}\right\}$
                Assign person $p$ to first free and available time periods with
                  duration $rd$ (construct $\mathbf{g}_{pit}$)
                $\tilde{\mathcal{I}} \leftarrow \tilde{\mathcal{I}} \setminus i$
            end
        end
    end
    Solve location subproblem for schedule $s$
end
```

6.1.2.4 Combine Schedules

In this section we elaborate on the cross-over operation to combine two schedules to form a new schedule. The algorithm starts by generating two scaled time periods t_1 and t_2, such that the number of time periods in between depends on the objective values of schedule s_1 and s_2. That is, the algorithm generates random t_1 at the begin of the time horizon, and sets t_2 based on t_1 and the scaled objective values of both schedules. The intuition in this step is that if one of the two schedules has desirable properties, then the algorithm shall construct the new schedule mostly based on this schedule. The genetic algorithm thus seeks to grasp the desirable properties in each iteration of the algorithm, while allowing for mutations of the schedule by random combing two schedules, having some randomness in generating time periods and apply mutation to each newly formed schedule as an attempt to overcome getting stuck in local optima. Using time periods t_1 and t_2, the new schedule is constructed by the approach as presented in Algorithm 3 below.

```
Algorithm 3: Combine Schedules
  Input: Schedules $s_1, s_2$
  Output: Schedule $s_{new}$
  $t_1 \sim \left\lceil U\left(0.5, -0.5 + \frac{\sum_{p\in\mathcal{P}}\sum_{i\in\mathcal{I}_p}\frac{\mathbf{Max}_{pi}}{|\mathcal{B}_p|}}{|\mathcal{P}|}\right)\right\rceil$
  $t' \leftarrow \frac{(z(s_1)-z(s^*))}{1+(z(s_1)-z(s^*))+(z(s_2)-z(s^*))}$
  $t_2 \leftarrow t_1 + \left\lceil \min\left\{\beta_1 \cdot |\mathcal{T}|,\ \max\{t',\ \beta_2 \cdot |\mathcal{T}|\}\right\}\right\rceil$
  for $p \in \mathcal{P}$ do
      Find last internship $i'$ person $p$ performs before time period $t_1$.
      Take order, duration and location assignment of internships person $p$
        performs in schedule $s_1$ up-to and including internship $i'$ in schedule $s_{new}$
      Find last internship $i''$ person $p$ performs before time period $t_2$.
      Take order, duration and location assignment of internships person $p$
        performs in schedule $s_2$ up-to and including internship $i''$ in schedule
        $s_{new}$ (excluding internships determined in previous step)
      Take order, duration and location assignment of internship person $p$
        performs in schedule $s_1$ (excluding internships determined in previous
        step)
      Construct schedule $s_{new}$ for person $p$ based on internship order, internship
        duration and location assignment of $s_{new}$
  end
```

Algorithm 3 is illustrated in Example 6.5 below.

Example 6.5. In this example we illustrate the cross-over operation for two people p_1 and p_2. Let the internships to be performed by p_1 and p_2 be: $\mathcal{I}_{p_1} = \mathcal{I}_{p_2} = \{1, \ldots, 6\}$. Let $\mathcal{L} = \mathcal{L}_i = \{1\}$, $\forall i \in \mathcal{I}$. Furthermore, let the order of internships for person p_1 and p_2 in schedule s_1 be (1, 2, 3, 4, 5, 6) and (2, 4, 3, 1, 5, 6), respectively. Let the order of internships for person p_1 and p_2 in schedule s_2 be (4, 3, 2, 1, 6, 5) and (1, 4, 2, 3, 5, 6), respectively. Then the cross-over operation results in schedule s_{new} as illustrated in Figure 4 below.

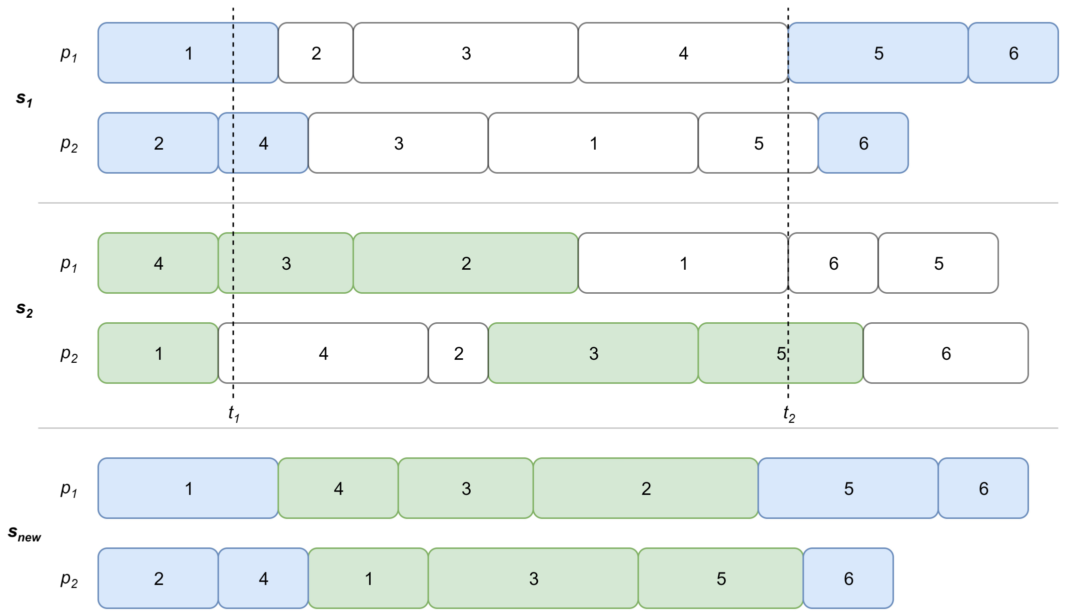


Figure 4: Illustration of the cross-over operation

6.1.2.5 Subproblem Greedy Approach

In this section we elaborate on a greedy approach to solve the location subproblem. For a given schedule s recall $\mathbf{g}_{pit}$ denotes if person p is scheduled during time period t to perform internship i and is considered fixed within the subproblem. This implies the objective function of the subproblem solely consists of penalties for over- and understaffing targets. The greedy algorithm starts with the initialization of the temporarily upper- and lower target, ξ_{ilt} and ζ_{ilt}, respectively. The algorithm then iteratively assign per person each of its internships to the location for which the remaining violation penalty is maximal. By assigning internships iteratively we ensure the algorithm performs favourably in terms of computation time, as the assignment of an internship is considered only ones during the algorithm. Furthermore, the aim of assigning an internship to the location with highest remaining penalty is to decrease the total violation penalty after each assignment. After each location assignment the temporary upper- and lower targets are updated. Lastly, we end the greedy algorithm by generating the schedule (uniquely given by x_{pilt}) and calculate the objective function $z(s)$ of the schedule. The greedy algorithm is summarized in Algorithm 4.

Algorithm 4: Subproblem Greedy Approach

```
Input: Current input schedule $s$, internship time periods $\mathbf{g}_{pit}$ of schedule $s$,
  internship lower- and upper targets ($\mathbf{LB}_{ilt}$ and $\mathbf{UB}_{ilt}$) and violation
  penalties ($\mathbf{c}_t^+$ and $\mathbf{c}_t^-$)
Output: Location assignment $v_{pil}$, representation of schedule $s$ as $x_{pilt}$ and
  objective function $z(s)$
$\xi_{ilt} \leftarrow \mathbf{UB}_{ilt}$
$\zeta_{ilt} \leftarrow \mathbf{LB}_{ilt}$
for $p \in \mathcal{P}$ do
    for $i \in \mathcal{I}_p$ do
        $l^* \leftarrow \arg\max_{l \in \mathcal{L}_i} \sum_{t \in \mathcal{T}_p \,:\, \mathbf{g}_{pit}=1} \Big\{ \mathbf{c}_t^+ \cdot \max\Big\{0,\ \xi_{ilt}\Big\} + \mathbf{c}_t^- \cdot \max\Big\{0,\ \zeta_{ilt}\Big\} \Big\}$
        $\xi_{il^*t} \leftarrow \xi_{il^*t} - \mathbf{g}_{pit}$
        $\zeta_{il^*t} \leftarrow \zeta_{il^*t} - \mathbf{g}_{pit}$
        $v_{pil^*} \leftarrow 1$
    end
end
$x_{pilt} \leftarrow \mathbf{g}_{pit} \cdot v_{pil}$
Calculate objective function $z(s)$ of current schedule $s$
```

6.1.2.6 Subproblem Exact Approach

In this section we elaborate on an exact approach to solve the location subproblem. For a given schedule s recall $\mathbf{g}_{pit}$ denotes if person p is scheduled during time period t to perform internship i, which is considered fixed within the subproblem. This implies the objective function of the subproblem solely consists of penalties corresponding to over- and understaffing targets. That is, the objective function is given by

$$\min \Bigg\{ \sum_{i \in \mathcal{I}} \sum_{l \in \mathcal{L}_i} \sum_{t \in \mathcal{T}} (\mathbf{c}_t^+ \cdot u_{ilt}^+ + \mathbf{c}_t^- \cdot u_{ilt}^-) \Bigg\}, \tag{6.15}$$

Such that

$$\sum_{l \in \mathcal{L}_i} v_{pil} = 1, \qquad \forall p \in \mathcal{P}, i \in \mathcal{I}_p, \tag{6.16}$$

$$\sum_{p \in \mathcal{P}} (\mathbf{g}_{pit} \cdot v_{pil}) - \mathbf{UB}_{ilt} \leq u_{ilt}^+, \qquad \forall i \in \mathcal{I}, l \in \mathcal{L}_i, t \in \mathcal{T}, \tag{6.17}$$

$$\mathbf{LB}_{ilt} - \sum_{p \in \mathcal{P}} (\mathbf{g}_{pit} \cdot v_{pil}) \leq u_{ilt}^-, \qquad \forall i \in \mathcal{I}, l \in \mathcal{L}_i, t \in \mathcal{T}, \tag{6.18}$$

$$v_{pil} \in \{0, 1\}, \qquad \forall p \in \mathcal{P}, i \in \mathcal{I}_p, l \in \mathcal{L}_i, \tag{6.19}$$

$$u_{ilt}^+, u_{ilt}^- \in \mathbb{N}_0, \qquad \forall i \in \mathcal{I}, l \in \mathcal{L}_i, t \in \mathcal{T}. \tag{6.20}$$

The objective function in (6.15) is to minimize the total violation penalty of over- and understaffing the location targets. Constraints (6.16) ensure that each internship of each person is assigned to exactly one location. Constraints (6.17) and (6.18) ensure that violations of the targets are added to the objective function. Constraints (6.19) and (6.20) ensure integrality of the decision variables.

The location subproblem involves symmetry in case for some internship i and two people p_1, p_2 such that $\forall t \in \mathcal{T}$ it holds that $\mathbf{g}_{p_1 it} = \mathbf{g}_{p_2 it}$ where for some solution to the subproblem it holds that $v_{p_1 i l_1} = 1$ and $v_{p_2 i l_2} = 1$ for two locations $l_1, l_2 \in \mathcal{L}_i : l_1 \neq l_2$. In that case, an alternative solution would be $v_{p_1 i l_2} = 1$ and $v_{p_2 i l_1} = 1$, resulting in the same objective value. Hence, to avoid symmetric solutions to the location subproblem we introduce the following symmetry breaking constraints

$$\sum_{l \in \mathcal{L}_i : l \leq l^*} v_{p_1 il} \geq \sum_{l \in \mathcal{L}_i : l \leq l^*} v_{p_2 il} \qquad \begin{array}{l} \forall p_1, p_2 \in \mathcal{P}, i \in \mathcal{I}_{p_1} \cap \mathcal{I}_{p_2}, l^* \in \mathcal{L}_i : \\ \forall t \in \mathcal{T}, \mathbf{g}_{p_1 it} = \mathbf{g}_{p_2 it},\ p_1 < p_2, \end{array} \tag{6.21}$$

Constraints (6.21) ensure that symmetric solutions to the location assignments $v_{p_1 il}, v_{p_2 il}$ are infeasible. That is, we restrict that, if for some internship i we have $\mathbf{g}_{p_1 it} = \mathbf{g}_{p_2 it}$ $\forall t \in \mathcal{T}$, the location assignment of this particular internship shall adhere to: p_1 is assigned to some location l_1 and person p_2 to some location l_2 such that $l_1 \leq l_2$.

6.1.2.7 Internship Duration Improvement Heuristic

In this section we elaborate on the internship duration improvement heuristic. We apply this heuristic in each iteration of the genetic algorithm to improve the quality of each schedule in the population. The purpose of this heuristic is to overcome potential non-fitting internship durations as constructed by the cross-over operation. That is, the cross-over operation (see Section 6.1.2.4) is specifically designed to create schedules with increasing quality regarding internship order, and hence there is an opportunity for improvement in terms of internship duration.

To satisfy this purpose, while simultaneously taking into account the requirement of a reasonable computation time, we consider only a subset of internship durations per person. We refer specifically to this subset as the set of evaluated internship durations. Assume person p performs internship i currently with duration cd_{pi}. Then, let the set of evaluated internship durations $\mathcal{ED}_{pi}$ in the improvement heuristic be defined as

$$\mathcal{ED}_{pi} = \left\{ ed \;\middle|\; \mathbf{Min}_{pi} \leq ed \leq \mathbf{Max}_{pi}, 0 \leq |ed - cd_{pi}| \leq \alpha \right\}. \tag{6.22}$$

Observe α denotes the distance between the current duration and set of evaluated durations. That is, α indicates the trade-off between computation time and potential quality improvement.

Next, we restrict the evaluation space even further by considering only a random subset of internships to evaluate per person. For each person p we restrict the algorithm to evaluate a maximum of ϕ out of $|\mathcal{I}_p|$ internships. Again, observe the trade-off between computation time and potential quality improvement is indicated by ϕ. The internship duration heuristic is summarized in Algorithm 5 below.

Algorithm 5: Internship Duration Improvement Heuristic

```
Input: Current schedule $s$ given by $x_{pilt}$, internship capacities ($\mathbf{UB}_{ilt}$ and
  $\mathbf{LB}_{ilt}$) and violation penalties ($\mathbf{c}_t^+$ and $\mathbf{c}_t^-$)
Output: Updated durations of internships of schedules $s$
$\mathcal{P}' \leftarrow \mathcal{P}$
$\xi_{ilt} \leftarrow \mathbf{UB}_{ilt} - \sum_{p \in \mathcal{P}} x_{pilt}$
$\zeta_{ilt} \leftarrow \mathbf{LB}_{ilt} - \sum_{p \in \mathcal{P}} x_{pilt}$
while $iterations \leq |\mathcal{P}|$ do
    Select random person $p \in \mathcal{P}'$
    $\mathcal{I}_p' \leftarrow \mathcal{I}_p$
    Update $\xi_{ilt}$ and $\zeta_{ilt}$
    while $|\mathcal{I}_p'| > |\mathcal{I}_p| - \phi$, $|\mathcal{I}_p'| > 0$ do
        Select random internship $i \in \mathcal{I}_p'$
        for $ed \in \mathcal{ED}_{pi}$ do
            Generate schedule for person $p$ using duration $ed$ for internship $i$
            Calculate objective $z_{ed}$ based on $\xi_{ilt}, \zeta_{ilt}$ and schedule of person $p$
        end
        Set duration of internship $i$ for person $p$ as $\underset{ed \in \mathcal{ED}_{pi}}{\arg\min}\, z_{ed}$
        Update $\xi_{ilt}$ and $\zeta_{ilt}$
        $\mathcal{I}_p' \leftarrow \mathcal{I}_p' \setminus i'$
    end
    $\mathcal{P}' \leftarrow \mathcal{P}' \setminus p$
end
Calculate objective $z(s)$ of current schedule $s$
```

6.1.2.8 Mathematical Programming Heuristic

In this section we elaborate on the mathematical programming heuristic used to improve the best schedule resulting from the genetic algorithm. The intuition of this heuristic is that we assume further improvement in terms of the duration of internships can be achieved with respect to order of internships and locations assignment. This heuristic exploits additional constraints imposed on the MIP formulation (see Section 5.5) to reduce the number of decision variables. These additional constraints ensure the order of internships and corresponding location assignments are fixed for each person. Note: an optimal solution to latter problem is not necessarily a global optimum. To restrict the location assignment we add the following constraints to the MIP formulation as given in Section 5.5

$$x_{pilt} \leq \mathbf{v}_{pil} \qquad \forall p \in \mathcal{P}, i \in \mathcal{I}_p, l \in \mathcal{L}_i, t \in \mathcal{T}_p. \tag{6.23}$$

Recall $\mathbf{v}_{pil}$ denotes the location assignment of internship i of person p of the given best schedule s_{best}. Hence, this constraint ensure the location assignment is fixed to the location assignment as was determined for schedule s_{best}. Also note that combining Constraints (5.13), (5.14) and (6.23) jointly ensure $\mathbf{v}_{pil} = 0$ implies $w_{pilt} = 0$.

For the second set of constraints we first introduce some notation. Let o_{ip} denote the order number of internship i for person p in the best schedule. That is, if person p performs internship i as its j-th internship, then $o_{ip} = j$. We use o_{ip} to define the set $\mathcal{O}_p$ as

$$\mathcal{O}_p = \Big\{(i', i'') \;\Big|\; \forall i', i'' \in \mathcal{I}_p,\ o_{i''p} - o_{i'p} = 1\Big\}. \tag{6.24}$$

Observe $\mathcal{O}_p$ is the set of ordered pairs of internships. That is, $\mathcal{O}_p$ consists of the internship pairs i', i'' such that person p performs internship i'' directly after internship i'. We illustrate $\mathcal{O}_p$ in Example 6.6 below.

Example 6.6. Let $\mathcal{I}_p = \{1, \ldots, 4\}$ and the number of blocks $b_{\max}$ be 1. Furthermore, let the order of internships for person p be (1, 2, 4, 3). That is, $o_{1p} = 1$, $o_{2p} = 2$, $o_{3p} = 4$ and $o_{4p} = 3$. Hence, $\mathcal{O}_p = \{(1,\ 2),\ (2,\ 4),\ (4,\ 3)\}$.

Next, we fix the order internships by imposing the following constraints to the MIP formulation

$$\sum_{l \in \mathcal{L}_{i'}} \sum_{t^* \in \mathcal{T}_p : t^* \leq t} w_{pi'lt^*} \geq \sum_{l \in \mathcal{L}_{i''}} w_{pi''lt} \qquad \forall p \in \mathcal{P}, (i', i'') \in \mathcal{O}_p, t \in \mathcal{T}_p. \tag{6.25}$$

Constraints (6.25) fix the order in which person p can start each internship during each time period t. Specifically, this constraint restricts the order of internships to the order as in schedule s_{best}.

The latter problem is solved by means of a branch-and-bound algorithm, which we warm-start using schedule s_{best}. The solution to the initial planning problem is then given by the resulting schedule from this branch-and-bound algorithm.

6.1.3 Population Diversity Measurement

In this section we elaborate on the diversity measurement of a population of schedules in the genetic algorithm. The aim of this measurement is to provide an alternative approach to compare the convergence of the population, rather than using objective values.

6.1.3.1 Schedule Diversity

Let $d(p_1, p_2)$ denote the distance between person p_1 in schedule s' and person p_2 in schedule s''. We calculate $d(p_1, p_2)$ as $d(p_1, p_2) = \sum_{i \in \mathcal{I}_{p_1}} \sum_{l \in \mathcal{L}_i} \sum_{t \in \mathcal{T}} |x_{p_1 ilt} - x_{p_2 ilt}|$, if the set of internships that shall be performed by p_1 and p_2 is equal (thus $\mathcal{I}_{p_1} = \mathcal{I}_{p_2}$). Let decision variable $a_{p_1 p_2} = 1$, if we link person p_1 from schedule s' to person p_2 of schedule s'', and be 0 otherwise.

6.1.3.2 Mathematical Formulation

In order to overcome potential symmetry between two schedules s_1 and s_2 we formulate the diversity measurement as a MIP problem. The objective function is

$$\min \left\{ \sum_{p_1 \in \mathcal{P}} \sum_{p_2 \in \mathcal{P}} d(p_1, p_2) \cdot a_{p_1, p_2} \right\}, \tag{6.26}$$

Such that

$$\sum_{p_1 \in \mathcal{P}} a_{p_1, p_2} = 1, \qquad \forall p_2 \in \mathcal{P}, \tag{6.27}$$

$$\sum_{p_2 \in \mathcal{P}} a_{p_1, p_2} = 1, \qquad \forall p_1 \in \mathcal{P}, \tag{6.28}$$

$$a_{p_1, p_2} \in \{0, 1\}, \qquad \forall p_1 \in \mathcal{P}, p_2 \in \mathcal{P} : \mathcal{I}_{p_1} = \mathcal{I}_{p_2}. \tag{6.29}$$

As the latter problem is a linear assignment problem the constraint matrix is totally unimodular, and hence the LP-relaxation is integer valued. Solving this problem for all pairs of schedules $s', s'' \in \mathcal{S} : s' < s''$ we obtain the total diversity.

6.2 Rescheduling Problem

In this section we elaborate on the methodology used to solve the rescheduling problem. First, we describe an exact approach in Section 6.2.1, which exhibits similarities to the exact approach used for the initial planning problem (see Section 6.1.1). Second, we elaborate on a heuristic approach in Section 6.2.2. Note: in this section we refer to the original schedule $\hat{x}_{pilt}$ as the schedule constructed using the order of internships, duration of internships and locations assignments corresponding to the schedule before the disruptive event is generated. This is necessary to overcome difficulties regarding availability changes of people. We use the original schedule $\hat{x}_{pilt}$ to calculate the rescheduling penalty as described in Section 5.6.

6.2.1 Exact Approach

In this section we elaborate on the exact approach to solve the rescheduling problem. The exact approach for the rescheduling problem consists of a branch-and-bound algorithm to solve the MIP formulation as described in Section 5. This formulation consists of the initial planning formulation (see Section 5.5) and the rescheduling extension (see Section 5.6). In Section 6.1.1.1 and 6.1.1.2 we discussed methods to improve the performance of the branch-and-bound algorithm. Furthermore, in Section 5.7.3 we proved the rescheduling problem is $\mathcal{NP}$-complete. Hence, we apply these improvement methods again to the rescheduling problem in order to reduce computation time and tighten the mathematical formulation.

6.2.2 Heuristic Approach

In this section we elaborate on the heuristic approach to solve the rescheduling problem. The heuristic approach is a variable neighborhood search approach. The intuition behind this approach is that, in contrast to the initial planning problem, the rescheduling is naturally more related to local search based strategies. That is, assuming the original schedule before disruptive events is close to optimality, we expect the optimal rescheduled schedule after disruptive events to be closely related to the original schedule. The reason for this is that we penalize any change made to the original schedule. Hence, we expect the rescheduled schedule mostly consists of similarities compared to the original schedule. To overcome getting stuck in a local optimum we exploit a variable neighborhood search strategy wherein each of the neighborhoods is randomly chosen. Thereby, each neighborhood is evaluated using a tabu search approach.

6.2.2.1 Relevant Notation

Before we describe an overview of the variable neighborhood search heuristic we first introduce the notation used in this approach:

- γ : number of iterations variable neighborhood search approach,
- $\mathcal{N}$: set of neighborhoods,
- w_n : weight of neighborhood n,

- $\tau_{pi'i''}$: iterations order swap of internship i' and i'' is on the tabu list for internship order swaps,
- ω_1 : iterations an internship order swap is tabu,
- $\psi_{i'i''}$: iterations internship duration change of internship i' and i'' is on the tabu list for duration changes,
- ω_2 : iterations a paired internship duration is tabu,
- $\mathcal{TD}_{pi'i''}$: set of paired durations to evaluate in the tabu search procedure for person p internship i' and i'',
- ξ_{ilt} : temporary upper target on the number of internships of type i at location l during time period t,
- ζ_{ilt} : temporary lower target on the number of internships of type i at location l during time period t,
- λ : temporary reschedule penalty after schedule change.

6.2.2.2 Variable Neighborhood Search Overview

We start the algorithm by initializing the original schedule. The algorithm consists of γ iterations of the variable neighborhood search heuristic. The first step in each iteration is to select a random neighborhood n'. This neighborhood is selected with probability $\frac{w_{n'}}{\sum_{n\in\mathcal{N}} w_n}$, where w_n denotes the weight of neighborhood n. Next, the algorithm performs a tabu search strategy in the selected neighborhood n'. That is, neighborhood n' is enumerated with respect to the current reschedule. We end each neighborhood by updating the current reschedule, the best solution and the corresponding tabu list to neighborhood n'. The neighborhoods used are provided in Section 6.2.2.3. Lastly, we solve the location subproblem in each iteration by means of an exact approach, on which we elaborate in Section 6.2.2.4. An overview of the heuristic approach is given in Algorithm 6 below.

Algorithm 6: Variable Neighborhood Search

```
Input: Original schedule
Output: Rescheduled schedule
Initialize original schedule
while $iterations \leq \gamma$ do
    Select random neighborhood based on weights
    Perform tabu search in neighborhood $n'$
    Solve subproblem with exact approach
end
```

6.2.2.3 Tabu Search Neighborhoods

We propose the following two neighborhoods in our variable neighborhood search heuristic:

1. **Tabu-Search 2-Opt Internship Swap.** In this neighborhood we compare all 2-opt swaps of two internships (for a fixed internship duration and location assignment) and apply the best swap to the schedule. This swap is then placed on the internship order tabu list given by $\tau_{pi'i''}$ for ω_1 iterations. This approach is summarized in Algorithm 7 below.

Algorithm 7: Tabu Search 2-Opt Internship Swap

```
Input: Schedule (represented by $x_{pilt}$)
Output: Updated schedule (represented by $x_{pilt}$)
$\xi_{ilt} \leftarrow \mathbf{UB}_{ilt} - \sum_{p \in \mathcal{P}} x_{pilt}$
$\zeta_{ilt} \leftarrow \mathbf{LB}_{ilt} - \sum_{p \in \mathcal{P}} x_{pilt}$
for $p \in \mathcal{P}$ do
    Update $\xi_{ilt}$ and $\zeta_{ilt}$
    for $b \in \mathcal{B}_p$ do
        for $i' \in \mathcal{I}_p \cap \mathcal{I}_b$ do
            for $i'' \in \mathcal{I}_p \cap \mathcal{I}_b \setminus i' \, : i' < i'',\ \tau_{pi'i''} = 0$ do
                Swap order of internships $i'$ and $i''$ for person $p$
                Construct schedule ($x_{pilt}$) for person $p$ based on swapped
                  internships, fixed location assignments and fixed
                  internship duration
                Update $\xi_{ilt}$ and $\zeta_{ilt}$
                $\lambda \leftarrow \sum_{i \in \mathcal{I}_p} \sum_{l \in \mathcal{L}_i} \sum_{t \in \mathcal{T}_p} \mathbf{d}_{pilt} \cdot \max\left\{0,\ x_{pilt} - \hat{x}_{pilt}\right\}$
                Calculate objective value based on $\xi_{ilt}, \zeta_{ilt}, \lambda$ and $x_{pilt}$
                Update $\xi_{ilt}$ and $\zeta_{ilt}$
            end
        end
    end
end
Do best 2-opt order swap for pair of internships $i'$ and $i''$ with best
  objective value, add pair of internships $i'$ and $i''$ to the order swap tabu
  list $\tau_{pi'i''}$ for $\omega_1$ iterations, update tabu list $\tau_{pi_1i_2}$ and best solution
```

2. **Tabu Search 2-Opt Internship Duration.** In this neighborhood we compare all 2-opt paired internship duration changes (for a fixed internship order and location assignment) and apply the best paired duration change to the reschedule. That is, we simultaneously change the duration of two internships for some person and place this paired duration change on the tabu list given by $\psi_{i'i''}$ for ω_2 iterations. For example, if person p has range on minimum and maximum duration for internship i_1 and i_2 given by $\{6, \ldots, 8\}$ and $\{10, \ldots, 11\}$

respectively, then $\mathcal{TD}_{i_1 i_2} = \{(6,10), (6,11), (7,10), (7,11), (8,10), (8,11)\}$. The algorithm evaluates each of these pairs $\forall p \in \mathcal{P}, b \in \mathcal{B}_p, i' \in \mathcal{I}_p \cap \mathcal{I}_b, i'' \in \mathcal{I}_p \cap \mathcal{I}_b : i' < i''$ and takes the pair with best objective value at the end of the algorithm. This approach is summarized in Algorithm 8.

Algorithm 8: Tabu Search 2-Opt Internship Duration

```
Input: Schedule (represented by $x_{pilt}$)
Output: Updated schedule (represented by $x_{pilt}$)
$\xi_{ilt} \leftarrow \mathbf{UB}_{ilt} - \sum_{p \in \mathcal{P}} x_{pilt}$
$\zeta_{ilt} \leftarrow \mathbf{LB}_{ilt} - \sum_{p \in \mathcal{P}} x_{pilt}$
for $p \in \mathcal{P}$ do
    Update $\xi_{ilt}$ and $\zeta_{ilt}$
    for $b \in \mathcal{B}_p$ do
        for $i' \in \mathcal{I}_p \cap \mathcal{I}_b$ do
            for $i'' \in \mathcal{I}_p \cap \mathcal{I}_b \setminus i' : i' < i'' : \psi_{i'i''} = 0$ do
                for $(td_{i'}, td_{i''}) \in \mathcal{TD}_{pi'i''}$ do
                    Change internship duration for internship $i'$ and $i''$ for
                      person $p$ to $(td_{i'}, td_{i''})$
                    Construct schedule $(x_{pilt})$ for person $p$ based on
                      changed internship duration for internship $i'$ and $i''$,
                      fixed location assignment and fixed internship order
                    Update $\xi_{ilt}$ and $\zeta_{ilt}$
                    $\lambda \leftarrow \sum_{i \in \mathcal{I}_p} \sum_{l \in \mathcal{L}_i} \sum_{t \in \mathcal{T}_p} \mathbf{d}_{pilt} \cdot \max\left\{0, x_{pilt} - \hat{x}_{pilt}\right\}$
                    Calculate objective value based on $\xi_{ilt}, \zeta_{ilt}, \lambda$ and $x_{pilt}$
                    Update $\xi_{ilt}$ and $\zeta_{ilt}$
                end
            end
        end
    end
end
Do best 2-opt internship duration for pair of internships $i'$ and $i''$ with
  best objective value, add pair of internships $i'$ and $i''$ to the internship
  duration change tabu list $\psi_{i'i''}$ for $\omega_2$ iterations, update tabu list $\psi_{i_1 i_2}$
  and best solution
```

Observe the variable neighborhood heuristic can easily be extended with additional neighborhoods to further improve the algorithms performance, but due to time limitations we stick to these two. Furthermore, note during each iteration of the variable neighborhood procedure only a single persons schedule is altered. That is, in the 2-opt internship order swap neighborhood we swap only two internships for one person and in the 2-opt internship duration neighborhood we change the duration of only two internships for one person.

6.2.2.4 Subproblem Exact Approach

The last step in each iteration of the variable neighborhood search algorithm is to solve the location subproblem by means of an exact approach. That is, we solve the location subproblem for a fixed order and duration of internships per person. The exact approach is similar as discussed in Section 6.1.2.6. However to incorporate rescheduling penalties we replace the objective given in (6.15) by the following expression

$$\min \left\{ \sum_{i \in \mathcal{I}} \sum_{l \in \mathcal{L}_i} \sum_{t \in \mathcal{T}} (\mathbf{c}_t^+ \cdot u_{ilt}^+ + \mathbf{c}_t^- \cdot u_{ilt}^-) + \sum_{p \in \mathcal{P}} \sum_{i \in \mathcal{I}_p} \sum_{l \in \mathcal{L}_i} \sum_{t \in \mathcal{T}_p} \mathbf{d}_{pilt} \cdot \mathbf{g}_{pit} \cdot v_{pil} \right\} \tag{6.30}$$

Observe the rescheduling penalty is given by the second term of this expression. Furthermore, we ensure feasibility of the rescheduled location assignments by imposing the following additional constraint to the formulation (see Section 6.1.2.6)

$$\mathbf{g}_{pit} \cdot v_{pil} \leq \hat{\mathbf{R}}_{pilt} \qquad \forall p \in \mathcal{P}, i \in \mathcal{I}_p, l \in \mathcal{L}_i, t \in \mathcal{T}_p. \tag{6.31}$$

7 Computational Experiments

In this section we elaborate on the computational experiments for the initial planning and rescheduling problem. We start with a description how each instance is generated. Thereafter, we elaborate on the parameter settings used to obtain results. The first results we discuss correspond to the initial planning problem. Next, we elaborate on the results for the rescheduling problem.

All computations were performed on a Windows computer with an Intel Core i7 6700HQ (2.60 GHz) processor and 8GB of RAM. We used CPLEX 12.9 to optimize the MIP problems and implemented the algorithms in AIMMS.

7.1 Instance Specifications

For each instance we provide the number of people in that instance p_{max}, the number of internships i_{max}, the number of blocks b_{max}, the number of locations l_{max} and the corresponding number of locations per internship type $|\mathcal{L}_i|$. We distributed p_{max} people linearly to start in each block. That is, at the start of the planning horizon each block has (up to rounding) the same number of people starting in each block. Furthermore, we ensure the different internship types are distributed linearly (up to rounding) over these blocks. Hence, for instances where $i_{max} \bmod b_{max} \neq 0$ some blocks have one internship more than others.

Given these characteristics, we generate for each internship type individually the range of durations from a discrete uniform distribution, such that the resulting range is a subset of $\{6, \ldots, 11\}$. We generate the range ($\{\mathbf{Min}_{pi}, \mathbf{Max}_{pi}\}$) for each person p for each internship type individually by taking the generated range for that internship type and add some small disturbance to the minimum and maximum of the given range based on a discrete uniform random variable which can take values $\{-1, \ldots, 1\}$.

Next, we generate the total capacity expected for each internship type i, as the number of internships in the corresponding block b_i divided by the number of people starting in block b_i To add some variety to the total capacity per internship type we multiply this by a uniformly distributed random variable on $(0.95, 1.05)$ individually generated for each internship type. Thereafter, we set the lower- and upper target for all time periods for some internship type i by distributing the total capacity random over $\mathcal{L}_i$ locations. That is, the lower- and upper target for each internship i and $\mathcal{L}_i$ remain unchanged over time ($\mathbf{UB}_{ilt} = \mathbf{LB}_{ilt}$, $\forall l \in \mathcal{L}_i, t \in \mathcal{T}$).

The penalties for staffing violations are set in consultation with someone that is involved in a practical way with this problem in a hospital traineeship program. The staffing penalties are set as

$$\mathbf{c}_t^+ = 1, \qquad \forall t \in \mathcal{T}, \tag{7.1}$$

$$\mathbf{c}_t^- = 2, \qquad \forall t \in \mathcal{T} : t \leq \sum_{p \in \mathcal{P}_{b1}} \sum_{i \in \mathcal{I}_{b1}} \left\{ \frac{\mathbf{MaxP}_{pi} + \mathbf{MinP}_{pi}}{2 \cdot |\mathcal{P}_{b1}|} \right\}, \tag{7.2}$$

$$\mathbf{c}_t^- = 0, \qquad \forall t \in \mathcal{T} : t > \sum_{p \in \mathcal{P}_{b1}} \sum_{i \in \mathcal{I}_{b1}} \left\{ \frac{\mathbf{MaxP}_{pi} + \mathbf{MinP}_{pi}}{2 \cdot |\mathcal{P}_{b1}|} \right\}. \tag{7.3}$$

The intuition here is that, hiring extra people (understaffing) is about twice as expensive as taking care of extra guidance (overstaffing) and shall therefore be penalized by a factor two. Also note the understaffing penalty is set to zero after some time period. This time period is set as the sum of the average of the durations corresponding to the first block, since after this time period understaffing cannot be prevented. That is, after this time period new people starting the traineeship in the first block are required to deal with understaffing. Next, the penalty corresponding to increasing the duration someones traineeship program is set to $\mathbf{b}_p = 0.30$. Hence, observe most emphasis is placed on minimizing staffing violations rather than program duration. Lastly, for the rescheduling problem we compare five settings of the rescheduling penalty in Figure 8 and 13 (where each reschedule option is allowed), after which the rescheduling penalty for the remaining rescheduling instances is set to $\mathbf{d}_{pilt} = 0.50$, and where we allowed all rescheduling options (that is, $\hat{\mathbf{R}}_{pilt} = 1$).

7.2 Parameter Settings

The algorithms presented in Section 6 require some parameter settings. All these parameters are set by extensive experimentation on several instances. We observed various parameter settings provided similar results. Due to the performance requirement in terms of computation time, we restricted computation time for the exact approach. Furthermore, we ensured to limit computation in the heuristic approach by experimenting only with parameters that led to reasonable computation times.

We set the number of iterations in the genetic algorithm θ to 100. The number of schedules constructed in the initial population is set to $\kappa = 50$. The parameters related to the rounding of t_2 in the cross over-operation, β_1 and β_2, are set to 0.10 and 0.90 respectively. In the duration improvement stage we set the parameters related to the trade-off between computation time and potential improvement, α and ϕ to 1 and 4 respectively.

In the variable neighborhood heuristic the number of iterations γ is set to 100. The number of iterations ω_1 and ω_2 a 2-opt internship swap and paired internship duration change are both set to 5. Neighborhood n is selected with weight $w_n = 1, \forall n \in \mathcal{N}$.

Next, for the initial planning problem we limited the computation time of the MIP approach to 18000 seconds, except for the results in which we study the different MIP improvement methods and the small instances in Table 1 and 15, where we used 1800 seconds per instance. The maximum computation time of the mathematical programming heuristic is set to 1800 seconds. Lastly, for the rescheduling problem we used a maximum of 7200 seconds for the MIP approach.

Lastly, to compare the genetic and variable neighborhood search algorithms we extended the number of iterations for the rescheduling algorithm to $\gamma = 300$ and applied the mathematical programming heuristic to both the genetic algorithm and the variable neighborhood search algorithm afterwards for a fair comparison.

7.3 Initial Planning Problem

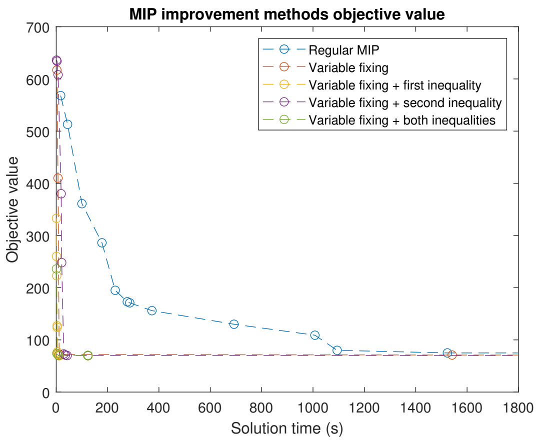
In this section we elaborate on the results obtained for the initial planning problem. We discuss results corresponding to the exact approach in Section 7.3.1. Second, we elaborate on the results obtained by the heuristic approach in Section 7.3.2. Due to time limitations we do not provide results for the population diversity measurement described in Section 6.1.3.

7.3.1 Exact Approach

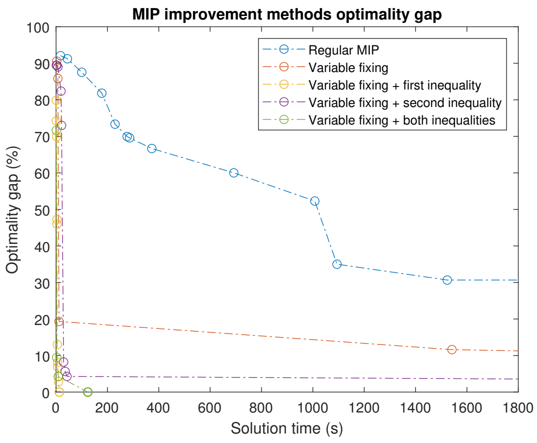
In this section we elaborate on the results obtained by the exact approach to solve the initial planning problem. In Figure 5 we show results for an instance of the initial planning problem with $p_{max} = 20$ people, $i_{max} = 4$ internships, $b_{max} = 2$ blocks, $l_{max} = 3$ locations and where for each internship i we have $|\mathcal{L}_i| = 2$ locations.

In Figure 5a the objective value of the best solution is provided for a given solution time. Figure 5b plots the optimality gap against solution time. We observe the MIP approach without improvement method is outperformed by each of the other improvement methods both in terms of objective value and optimality gap. In Figure 5 we provide the same instance in more detail. Here, we observe the improvement method corresponding to variable fixing plus the first valid inequality (6.13) and variable fixing plus both inequalities (6.13) and (6.14) perform best compared to the other improvement methods and perform similarly if compared to each other. Lastly, we find it notable the improvement method with variable fixing plus the second inequality (6.14) is not performing as good as the improvement methods with the first valid inequality. Hence, we conclude the best approach is to exploit the variable fixing methods plus the first valid inequality.

In Appendix A we provide additional results for an instance with $p_{max} = 30$ in Figure 9 and 10, and for an instance with $i_{max} = 5$ in Figure 11 and 12. Both instances led to the same conclusions. We shall note that model building time is excluded in these plots, since this was negligible compared to solution time (in these instances).

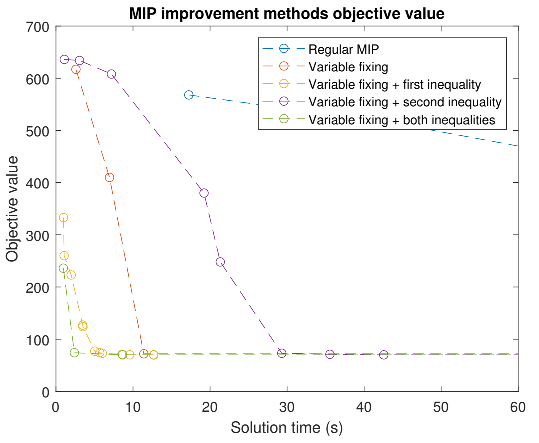


(a) Objective value.

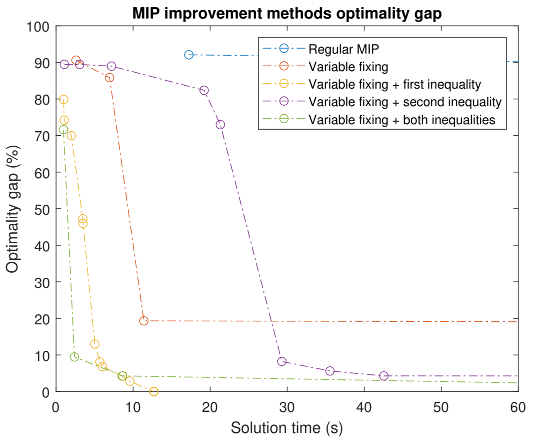


(b) Optimality gap.

Figure 5: Effect of MIP improvement methods.



(a) Objective value.



(b) Optimality gap.

Figure 6: Effect of MIP improvement methods detailed.

7.3.2 Heuristic Approach

In this section we discuss results related to the initial planning problem. First, we compare in Section 7.3.2.1 the heuristic approach with the MIP approach, where we applied variable fixing and both valid inequalities. Next, we zoom into the location subproblem in Section 7.3.2.2, where we compare the greedy heuristic against the exact approach and the exact approach with symmetry breaking constraints. We elaborate on the mathematical programming heuristic in Section 7.3.2.3. Lastly, in Section 7.3.2.4 we apply the initial planning algorithm and rescheduling algorithm for an instance of the initial planning problem and compare both algorithms in terms of solution quality and computation time.

7.3.2.1 Genetic Algorithm

Table 1 provides results for the initial planning problem for both the MIP and the heuristic approach. Each row in this table corresponds to one instance of the initial planning problem with p_{max} people. Furthermore, in each instance we have $i_{max} = 3$ internships $b_{max} = 1$, blocks, $l_{max} = 2$ locations and $|\mathcal{L}_i|= 2$ locations per internship type. The optimality gap for the MIP and heuristic approach are given by Gap MIP (%) and Gap H (%) respectively and are computed using the LP-relaxation lowerbound where we applied variable fixing and valid inequalities. In Table 15 in Appendix A a similar experiment is provided, where the number of internships is $i_{max} = 4$, $b_{max} = 2$ blocks, $l_{max} = 2$ locations and $|\mathcal{L}_i|= 2$ locations per internship type.

Table 1: Initial planning problem
$i_{max} = 3,\ b_{max} = 1, l_{max} = 2,\ |\mathcal{L}_i|= 2.$

p_{max}	Gap MIP (%)	Gap H (%)
4	0.00	0.00
5	0.00	3.92
6	0.00	0.00
7	5.13	5.41
8	0.00	0.00
9	0.00	0.00
10	0.00	0.00
11	5.09	5.37
12	12.96	12.77
13	1.67	1.69
14	1.75	1.79
15	0.00	3.92
16	9.21	9.42
17	3.03	6.25
18	0.00	0.00

In this table we provide results for the initial planning problem for instances with $i_{max} = 3$ internships, $b_{max} = 1$ blocks, $l_{max} = 2$ locations and $|\mathcal{L}_i|= 2$ locations per internship. The number of people is given by p_{max} and corresponds to one instance. The optimality gap for the MIP and heuristic approach are given by Gap MIP (%) and Gap H (%), respectively. The solution time for the MIP approach was restricted to a maximum of 1800 seconds per instance.

From Table 1 we conclude the heuristic and MIP approach perform approximately similar in terms of solution quality. However, we note the solution time for the MIP approach was limited to 1800 seconds, whereas the heuristic required at most 150 seconds per instance. Furthermore, we observe that for the instance with $p_{max} = 5$, $p_{max} = 15$ and $p_{max} = 17$ the MIP approach outperforms the heuristic in terms of optimality. Analyzing these instances in more depth reveals the total staffing penalty was particularly low in these instances compared to the other instances. That is, in absolute terms the objective values were reasonably comparable. Especially, the number of staffing violations was significantly higher in the other instances.

Next, Table 2 provides results for larger instances compared to Table 1 Each row corresponds to one instance with a given number of people p_{max}, $i_{max} = 8$ internships $b_{max} = 3$ blocks, $l_{max} = 5$ locations and $|\mathcal{L}_i|= 3$ locations per internship type. The optimality gap for the heuristic approach is given by Gap H (%) and is computed using the LP-relaxation lowerbound where we applied variable fixing and valid inequalities.

Table 2: Initial planning problem
$i_{max} = 8$, $b_{max} = 3$, $l_{max} = 5$, $|\mathcal{L}_i|= 3$.

p_{max}	Time H (s)	Time MIP (s)	Gap H (%)
10	842	9	0.00
15	1334	3789	1.06
20	1606	>18000	17.67
25	2053	>18000	6.24
30	3579	>18000	19.43
35	3890	>18000	5.30
40	3601	>18000	6.85
45	4789	>18000	7.07
50	4796	>18000	10.58
55	5721	>18000	5.68
60	5733	>18000	11.60
65	7712	>18000	7.89
70	8410	>18000	22.36
75	8665	>18000	3.02
80	8454	>18000	10.48

In this table we provide results for the initial planning problem for instances with $i_{max} = 8$ internships, $b_{max} = 3$ blocks, $l_{max} = 5$ locations and $|\mathcal{L}_i|= 3$ locations per internship. The number of people is given by p_{max} and corresponds to one instance. The optimality gap for the heuristic approach is given by Gap H (%). The solution times for the heuristic and MIP approach are given by Time H and Time MIP, respectively. For the instances with $p_{max} \geq 20$ the MIP approach was not able to find a feasible solution within the restricted maximum computation time of 18000 seconds per instance. For the instances $p_{max} = 10$ and $p_{max} = 15$ the MIP was solved to optimality.

From Table 2 we conclude the heuristic outperforms the MIP approach, since the MIP approach is not able to find feasible solutions for instances with $p_{max} \geq 20$. Furthermore, we find it notable the optimality gap is varying significantly. This could be caused by the fact we only optimized one instance for a given number of people p_{max}. Also note the optimality gap was computed using the LP-relaxation lowerbound, which is likely to be inaccurate for these larger instances (compared to the instances in Table 1). Another important observation is that the MIP approach is not able to find a feasible solution within 18000 seconds and hence the MIP approach cannot be used for practical problem.

Lastly, we show results for even larger instances (compared to the instances in Table 1 and 2) of the initial planning problem in Table 3 with $i_{max} = 12$ internships, $b_{max} = 4$ blocks, $l_{max} = 3$ locations and $|\mathcal{L}_i| = 2$ locations per internship type.

Table 3: Initial planning problem
$i_{max} = 12$, $b_{max} = 4$, $l_{max} = 3$, $|\mathcal{L}_i| = 2$.

p_{max}	Time H (s)	Time MIP (s)	Gap H (%)
20	2694	>18000	0.46
20	2429	>18000	0.78
20	3431	>18000	1.89
30	5537	>18000	17.48
30	5301	>18000	25.62
30	5450	>18000	14.53
55	8641	>18000	26.56
55	9055	>18000	16.38
55	8831	>18000	25.35
80	10460	>18000	25.64
80	10496	>18000	21.33
80	10898	>18000	31.88
100	11328	>18000	23.56
100	10967	>18000	22.00
100	11136	>18000	22.89

In this table we provide results for the initial planning problem for instances with $i_{max} = 12$ internships, $b_{max} = 4$ blocks, $l_{max} = 3$ locations and $|\mathcal{L}_i| = 2$ locations per internship. The number of people is given by p_{max} and corresponds to one instance. The optimality gap for the heuristic approach is given by Gap H (%). The solution times for the heuristic and MIP approach are given by Time H and Time MIP, respectively. In all instances the MIP approach was not able to find a feasible solution within the restricted maximum computation time of 18000 seconds per instance.

Again, we see in Table 3 the MIP approach is not able to find a feasible solution within 18000 seconds. This result is as expected since we have more internships in this instance and also more internships per block, which increases the potential number of orders each person can perform these internships. As we observed in Table 2 the optimality gap seems varying for several instances. Hence, in Table 3 we provide results for three different instances with the same characteristics. For several pairs of these three instances we again observe high variations in the optimality gap. This could be caused by the fact that the lowerbound is computed using the LP-relaxation, which is probably inaccurate for larger instances.

Lastly, we observe the computation time for the heuristic approach seems to flatten, which is partially caused by the fact that for larger instances the mathematical programming heuristic requires the maximum computation time of 1800 seconds in contrast to the smaller instances with $p_{max} = 20$ and $p_{max} = 30$ for which the mathematical programming heuristic terminated because of optimality. That is, if we would increase the maximum computation time for the mathematical programming heuristic the computation time of the heuristic approach would also increase significantly.

7.3.2.2 Location Subproblem

In this section we compare the greedy heuristic described in Section 6.1.2.5 to the exact approach described in Section 6.1.2.6 to solve the location subproblem. For the exact approach we provide results for both including and excluding symmetry breaking constraints.

In Table 4 − 7 we provide results for the location subproblem. For each table we provide the number of internships i_{max}, the number of blocks b_{max}, the number of locations per internship type $|\mathcal{L}_i|$. In each table we vary the number of people p_{max} and the total number of locations l_{max}. Each row in these tables corresponds to 5000 instances for which we provide the average optimality gap for the greedy heuristic by Gap H (%), the total cumulative computation time by Time H (s), Time MIP (s) and Time MIP SYM (s), for the greedy heuristic, MIP approach excluding symmetry breaking constraints and including symmetry breaking constraints respectively.

Each instance was obtained by applying the genetic algorithm to the problem, where we continued the algorithm with the solution provided by the exact approach without symmetry breaking. That is, during each iteration of the genetic algorithm we solved the location subproblem with each approach for each newly constructed schedule.

Table 4: Location subproblem comparison of greedy heuristic, exact approach excluding symmetry breaking constraint and exact approach including symmetry breaking
$i_{max} = 5$, $b_{max} = 2$, $|\mathcal{L}_i| = 2, \forall i \in \mathcal{I}$.

l_{max}	p_{max}	Gap H (%)	Time H (s)	Time MIP (s)	Time MIP SYM (s)
2	30	0.78	122	239	259
2	70	0.54	215	309	340
2	100	0.31	309	360	402
3	30	0.63	114	224	248
3	70	0.18	207	292	326
3	100	0.23	277	341	385
4	30	0.75	108	222	241
4	70	0.57	229	319	356
4	100	0.21	304	365	409
5	30	1.11	134	252	274
5	70	0.62	238	335	367
5	100	0.59	357	415	458

In this table results are presented for the location subproblem. In each instance of the problem we have $i_{max} = 5$ internships , $b_{max} = 2$ blocks , $|\mathcal{L}_i| = 2, \forall i \in \mathcal{I}$ locations per internship. The number of people is given by p_{max}. The total number of locations is given by l_{max}. Each row in this table corresponds to a total of 5000 instances of the problem solved throughout the genetic algorithm. For the greedy heuristic we provide the average optimality gap (Gap H (%)). The total cumulative computation time each of the three approaches is given by Time H(s), Time MIP (s) and Time MIP SYM (s), for the greedy heuristic, exact approach excluding symmetry breaking and including symmetry breaking respectively.

Table 5: Location subproblem comparison of greedy heuristic, exact approach excluding symmetry breaking constraint and exact approach including symmetry breaking
$i_{max} = 12$, $b_{max} = 3$, $|\mathcal{L}_i| = 2, \forall i \in \mathcal{I}$.

l_{max}	p_{max}	Gap H (%)	Time H (s)	Time MIP (s)	Time MIP SYM (s)
2	30	3.90	174	440	493
2	70	2.82	357	548	628
2	100	1.92	529	662	771
3	30	3.37	181	428	496
3	70	3.21	370	561	653
3	100	2.02	530	671	787
4	30	3.51	182	427	479
4	70	2.46	370	553	652
4	100	1.20	534	664	777
5	30	4.02	183	444	506
5	70	2.11	377	568	652
5	100	1.08	524	684	797

In this table results are presented for the location subproblem. In each instance of the problem we have $i_{max} = 12$ internships , $b_{max} = 3$ blocks , $|\mathcal{L}_i| = 2, \forall i \in \mathcal{I}$ locations per internship. The number of people is given by p_{max}. The total number of locations is given by l_{max}. Each row in this table corresponds to a total of 5000 instances of the problem solved throughout the genetic algorithm. For the greedy heuristic we provide the average optimality gap (Gap H (%)). The total cumulative computation time each of the three approaches is given by Time H(s), Time MIP (s) and Time MIP SYM (s), for the greedy heuristic, exact approach excluding symmetry breaking and including symmetry breaking respectively.

Table 6: Location subproblem comparison of greedy heuristic, exact approach excluding symmetry breaking constraint and exact approach including symmetry breaking
$i_{max} = 7$, $b_{max} = 3$, $|\mathcal{L}_i| = 3, \forall i \in \mathcal{I}$.

l_{max}	p_{max}	Gap H (%)	Time H (s)	Time MIP (s)	Time MIP SYM (s)
3	30	1.08	115	292	352
3	70	0.75	233	419	511
3	100	1.15	310	473	594
4	30	2.04	130	321	376
4	70	1.41	259	445	539
4	100	2.29	367	558	681
5	30	3.30	137	329	386
5	70	1.48	227	401	499
5	100	1.30	324	511	641
6	30	1.65	121	289	350
6	70	1.45	236	404	492
6	100	1.03	320	500	627

In this table results are presented for the location subproblem. In each instance of the problem we have $i_{max} = 7$ internships , $b_{max} = 3$ blocks , $|\mathcal{L}_i| = 3, \forall i \in \mathcal{I}$ locations per internship. The number of people is given by p_{max}. The total number of locations is given by l_{max}. Each row in this table corresponds to a total of 5000 instances of the problem solved throughout the genetic algorithm. For the greedy heuristic we provide the average optimality gap (Gap H (%)). The total cumulative computation time each of the three approaches is given by Time H(s), Time MIP (s) and Time MIP SYM (s), for the greedy heuristic, exact approach excluding symmetry breaking and including symmetry breaking respectively.

Table 7: Location subproblem comparison of greedy heuristic, exact approach excluding symmetry breaking constraint and exact approach including symmetry breaking
$i_{max} = 12$, $b_{max} = 3$, $|\mathcal{L}_i| = 3, \forall i \in \mathcal{I}$.

l_{max}	p_{max}	Gap H (%)	Time H (s)	Time MIP (s)	Time MIP SYM (s)
3	30	5.50	212	606	726
3	70	4.28	416	769	929
3	100	2.79	585	1026	1224
4	30	5.15	213	591	716
4	70	4.07	423	796	955
4	100	2.88	592	1071	1271
5	30	5.89	223	622	743
5	70	4.28	431	796	954
5	100	3.35	600	1098	1295
6	30	6.68	226	639	758
6	70	2.48	429	812	976
6	100	3.13	581	1066	1270

In this table results are presented for the location subproblem. In each instance of the problem we have $i_{max} = 12$ internships , $b_{max} = 3$ blocks , $|\mathcal{L}_i| = 3, \forall i \in \mathcal{I}$ locations per internship. The number of people is given by p_{max}. The total number of locations is given by l_{max}. Each row in this table corresponds to a total of 5000 instances of the problem solved throughout the genetic algorithm. For the greedy heuristic we provide the average optimality gap (Gap H (%)). The total cumulative computation time each of the three approaches is given by Time H(s), Time MIP (s) and Time MIP SYM (s), for the greedy heuristic, exact approach excluding symmetry breaking and including symmetry breaking respectively.

Based on the results for the location subproblem provided in Table 4 − 7 some observations and conclusions can be made. First, comparing both exact approaches we observe the exact approach outperforms the exact approach with symmetry breaking constraints. At first sight this result seems counter-intuitive since breaking symmetry in a branch-and-bound procedure generates extra cuts which normally results in a reduction of computation time. However, a deeper analysis reveals that indeed the solution time of the branch-and-bound procedure is decreased for the MIP approach with symmetry breaking constraints, but that the additional model building time caused by these constraints is larger than the reduction in solution time. Hence, we conclude that symmetry breaking shall not be applied in the location subproblem when an exact approach is used.

Next, we observe the greedy heuristic is significantly faster than the exact approach while finding high-quality solutions for all instances. Here, we shall emphasize that the shown optimality gaps are not present in the final solution in the genetic algorithm as we apply the exact approach in the last iteration of the algorithm to overcome this problem. However, the selection of schedules in the genetic algorithm does not require an optimal solution to the location subproblem, but only an indication whether a certain schedule is reasonably good to maintain in the next iteration. This supports the fact that we used the greedy heuristic in the genetic algorithm to decrease solution time, while ensuring optimal location assignment (with respect to. internship order and duration) in the last iteration of the algorithm.

7.3.2.3 Mathematical Programming Heuristic

In this section we elaborate on the mathematical programming heuristic. In Figure 7 we provide a comparison of the objective value before and after the mathematical programming heuristic was applied. In these instances we used $p_{max} = 30$ number of people, $i_{max} = 12$ internships, $b_{max} = 4$ blocks, $l_{max} = 3$ locations and $|\mathcal{L}_i| = 2$ locations per internship type. Thereby, the computation time for each of the 250 instances was set to a maximum of 1800 seconds. This result is obtained by applying the mathematical programming heuristic in each iteration of the genetic algorithm to each schedule in the population before the selection procedure is applied.

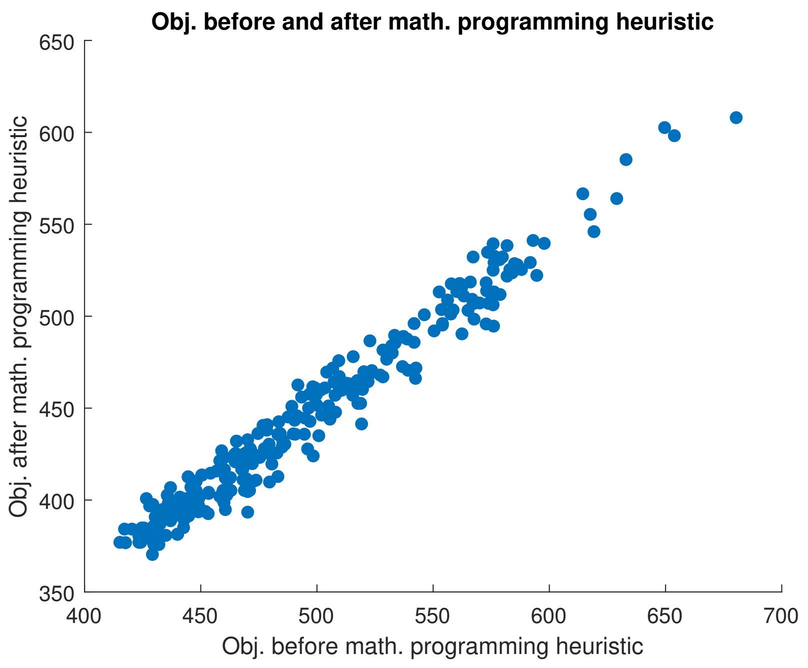


Figure 7: Improvement of mathematical programming heuristic.

An important observation is the almost linear relationship between objective values before and after the mathematical programming heuristic is applied. This supports the fact that we select schedules based on objective value and apply the heuristic to the best schedule resulting from the genetic algorithm. Another observation is the significant improvement the heuristic achieves, which is 9.81% on average for these specific instances. Thereby, we shall note that in the experimentation phase of this heuristic we observed the best approach is to apply the mathematical programming heuristic to a single schedule, rather than distributing the same solution time over several schedules.

7.3.2.4 Algorithm Comparison

In this section we compare the initial planning algorithm and the rescheduling algorithm for instances of the initial planning problem. The characteristics of the instances generated for this comparison are given by the scenarios in Table 8 below. For each scenario we provide the number of internships i_{max}, the number of of blocks b_{max}, the number of locations l_{max} and the number of locations per internship type $|\mathcal{L}_i|$.

Table 8: Scenarios initial planning problem for algorithm comparison.

Scenario	i_{max}	b_{max}	l_{max}	$\lvert\mathcal{L}_i\rvert$
1	8	2	5	3
2	12	4	5	4

In this table we provide the characteristics for the two scenarios used to compare the initial planning and rescheduling algorithm for instances of the initial planning problem. For each scenario we provide the number of internships i_{max}, the number of of blocks b_{max}, the number of locations l_{max} and the number of locations per internship type $|\mathcal{L}_i|$.

In Table 9 we compare the initial planning algorithm and the rescheduling algorithm. For a given number of people p_{max} we generate five instances based on the given scenario. We provide the average computation time and average optimality gap Time I(s), Time R(s), Gap I (%) and Gap R (%) for the initial planning and rescheduling algorithm respectively.

Table 9: Initial planning problem algorithm comparison

Scenario	p_{max}	Time I (s)	Time R (s)	Gap I (%)	Gap R (%)
1	20	1710	1520	1.35	1.32
	40	3902	4217	15.11	18.93
	80	9721	10923	22.96	27.38
2	20	2780	2408	3.37	3.90
	40	7534	8405	18.25	23.12
	80	17590	22871	24.09	29.59

In this table we compare the initial planning and rescheduling algorithm for instances of the initial planning problem. Each row corresponds to five instances generated in accordance to the specifications of the corresponding scenario as described in Table 8. The number of people is given by p_{max}. We provide the average computation time and optimality gap Time I(s), Time R(s), Gap I (%) and Gap R (%) for the initial planning and rescheduling algorithm respectively.

Based on the results presented in Table 9 we conclude that the initial planning algorithm outperforms the rescheduling algorithm for solving the initial planning problem. We observe the initial planning problem finds significantly better solutions than the rescheduling algorithm, especially for medium and larger instances. Also note the increasing gap between Gap I and Gap R, indicating the initial planning algorithm is increasingly better than the rescheduling algorithm when the instance size increases.

7.4 Rescheduling Problem

In this section we elaborate on the rescheduling problem. First, we provide some insight in Section 7.4.1 regarding the trade-off between the rescheduling penalty, staffing penalty at locations and the number of changes made to the original schedule. Thereafter, we discuss results corresponding to single event rescheduling in Section 7.4.2. Recall the original schedule $\hat{x}_{pilt}$ to calculate the rescheduling penalty, is the schedule constructed using the order of internships, duration of internships and locations assignments corresponding to the schedule before disruptive events are generated. Lastly, we elaborate on a simulation setting wherein multiple events are generated over time.

7.4.1 Penalty Trade-off

In this section we elaborate on the trade-off regarding the setting of the rescheduling penalty and the resulting staffing penalty for violating internship targets at locations and the number of changes made to the original schedule. In Figure 8 we illustrate this trade-off for five settings of the rescheduling penalty $\mathbf{d}_{pilt} \in \{0, 0.25, 0.50, 0.75, 1\}$. In this particular instance we used $p_{max} = 40$ people, $i_{max} = 6$ internships, $b_{max} = 3$ blocks, $l_{max} = 3$ locations where each internship can be performed at $|\mathcal{L}_i| = 2$ locations. For each individual setting of the rescheduling penalty we generated five instances, which are optimized using the MIP approach. In each instance we change the availability of five randomly chosen people. For each person we change the availability starting from time period 2 up to a random generated time period t_{random}, where t_{random} is generated using a discrete uniform distribution on the interval $(6, 10)$.

On the left-hand side of this figure we provide the average number of changes made to the original schedule. On the right-hand side the corresponding average location penalty incurred for violating staffing targets. A similar experiment is given in Figure 13 in Appendix C, where the number of people p_{max} is set to 80 and the number of people, for which availability changes are generated, to ten.

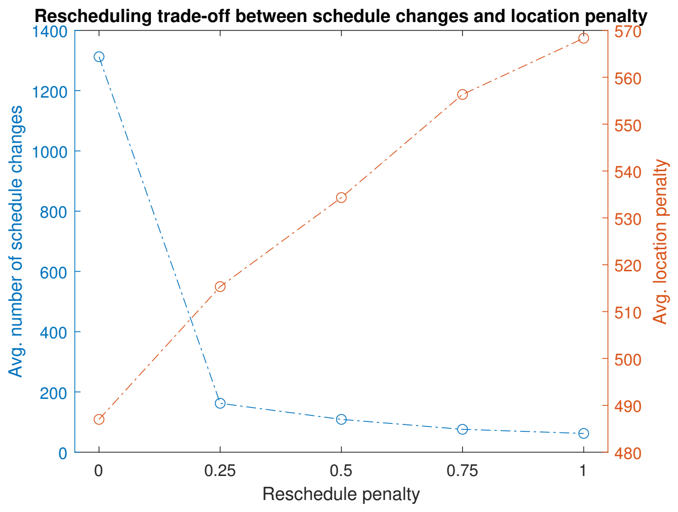


Figure 8: Rescheduling trade-off instances with $p_{max} = 40$ people.

An important observation is the significant high number of changes made to the original schedule for a rescheduling penalty of $\mathbf{d}_{pilt} = 0$. Second, we see the average location penalty to be negatively related to the number of schedule changes. This result shows the trade-off between the number of changes made to the schedule and Based on these observations we decided to use a rescheduling penalty of $\mathbf{d}_{pilt} = 0.50$ in our remaining rescheduling experiments.

7.4.2 Single Event Rescheduling

In this section we elaborate on single event rescheduling problems. In Table 11 $-$ 13 we provide results for rescheduling problems corresponding to availability changes of people, staffing target changes at locations and additional people joining the traineeship program, respectively. For these rescheduling experiments we used two scenarios as presented in Table 10, where the number of internships is given by i_max, the number of blocks by b_{max}, the number of locations by l_{max} and the number of locations per internship type by $|\mathcal{L}_i|$. Thereby, in Table 11 $-$ 13 we provide the number of people p_{max}, the average computation time of the heuristic approach and exact approach by Time H (s) and Time MIP (s) and the average optimality gap of the heuristic approach Gap H $(\%)$. Each row in Table 11 $-$ 13 corresponds to five experiments of the rescheduling problem where the same type of events are generated.

For each instance of the rescheduling problem we first create an initial planning by solving the initial planning problem using the genetic algorithm and thereafter generated the given disruptive events. In Table 11 we provide results for the rescheduling problem corresponding to availability changes of people. In each instance we change the availability of five randomly chosen people. For each person we change the availability starting from time period 2 up to a random generated time period t_{random}, where t_{random} is generated using a discrete uniform distribution on the interval $(6, 10)$. In Table 12 we provide results for the rescheduling problem where the staffing target is altered. Here, we generate for each internship a random variable based on a discrete uniform distribution on the interval $(-1, 1)$, $(-2, 2)$ and $(-4, 4)$ for p_{max} is 20, 40 and 80 respectively, which is then added to the staffing targets $\mathbf{UB}_{ilt}$ and $\mathbf{LB}_{ilt}$. In Table 13 we added five people to the original planning at the first time period t for which $\mathbf{c}_t^- = 0$ (see Equation (7.3)) and then set $\mathbf{c}_t^- = 2$ as in Equation (7.2) and update the corresponding staffing targets. That is, we alter the staffing targets as described in Section 7.1.

Table 10: Scenarios rescheduling planning problem.

Scenario	i_{max}	b_{max}	l_{max}	$\|\mathcal{L}_i\|$
1	8	2	5	3
2	12	4	5	4

In this table we provide the characteristics for two scenarios that are used to generate instances of the rescheduling problem. For each scenario we provide the number of internships i_{max}, the number of of blocks b_{max}, the number of locations l_{max} and the number of locations per internship type $|\mathcal{L}_i|$.

Table 11: Single event rescheduling problem: Availability change.

Scenario	p_{max}	Time H (s)	Time MIP (s)	Gap H (%)
1	20	180	95	4.20
	40	350	480	6.06
	80	746	3902	5.99
2	20	340	211	3.57
	40	630	974	4.33
	80	1214	6704	3.84

In this table we provide results related to the rescheduling problem where the availability of people is changed. Each row in this table corresponds to five instances, for which we provide the average results. We compare the variable neighborhood search heuristic and MIP approach computation time which is given by Time H (s) and Time MIP (s), respectively. The optimality gap of the variable neighborhood search heuristic is given by Gap H (%).

Table 12: Single event rescheduling problem: Staffing target change.

Scenario	p_{max}	Time H (s)	Time MIP (s)	Gap H (%)
1	20	191	110	5.47
	40	349	425	4.24
	80	812	3259	4.69
2	20	329	232	4.22
	40	593	771	3.91
	80	1198	6534	4.10

In this table we provide results related to the rescheduling problem where the staffing target are changed. Each row in this table corresponds to five instances, for which we provide the average results. We compare the variable neighborhood search heuristic and MIP approach computation time which is given by Time H (s) and Time MIP (s), respectively. The optimality gap of the variable neighborhood search heuristic is given by Gap H (%).

Table 13: Single event rescheduling problem: Additional people.

Scenario	p_{max}	Time H (s)	Time MIP (s)	Gap H (%)
1	20	209	3957	10.12
	40	402	>7200*	8.24
	80	898	>7200*	6.02
2	20	436	>7200*	7.90
	40	850	>7200*	6.34
	80	1803	>7200*	5.97

In this table we provide results related to the rescheduling problem where five additional people are included in the traineeship program. Each row in this table corresponds to five instances, for which we provide the average results. We compare the variable neighborhood search heuristic and MIP approach computation time which is given by Time H (s) and Time MIP (s), respectively. We indicate the MIP approach was not solved to optimality by * after 7200 seconds of computation time. The optimality gap of the variable neighborhood search heuristic is given by Gap H (%).

Based on the results in Table 11 − 13 some conclusions can be drawn. First, we observe in Table 11 and 12 the MIP approach is able to find the optimal solution within 7200 seconds of computation This result seems counter-intuitive since we observed for the initial planning problem the MIP approach was not able to find a feasible solution for instances with a similar size. However, the difference with the initial planning problem is the rescheduling penalty, which results in different type of optimization problem since any change to the original planning is penalized.

This is inline with the results described in Table 13, where five additional people are added to the program. For these additional people we used a rescheduling of 0, resulting in a mixed setting of the initial planning and rescheduling problem. Also observe the optimality are decreasing for increasing instance sizes in Table 13. This is caused by the decreasing effect of five additional people to an increasing number of people in the original instance given by p_{max}. That is, the effect of five additional to an instance with $p_{max} = 20$ people is higher than on an instance with $p_{max} = 80$ people. Hence, we conclude that for rescheduling purposes related to additional people events the heuristic outperforms the MIP approach, especially for larger instances.

7.4.3 Simulation Setting

In this section we elaborate on a simulation setting of the problem. For this purpose, we generated six events at time periods $\{6, 9, 12, 16, 20, 24\}$ where each of these events is generated in a similar fashion as described in Section 7.4.2, but now starting the event at the given time period. The following events are generated: AC, TC and AP, corresponding to availability changes of people, staffing target changes at locations and additional people added to the program respectively.

In Table 14 we provide the average results of five instances evaluated over a time horizon of 24 time periods. We provide the cumulative penalty for the MIP and heuristic (H) approach up-to and including the time period the event is generated. For example, the cumulative location penalty corresponding to the first event (availability change generated at time period $t = 6$) is calculated as the cumulative location penalty up-to and including time period $t = 6$. Next, we provide the total rescheduling penalty incurred by both approaches based on the rescheduled planning at the generated event. For example, the rescheduling penalty corresponding to the second event (target change generated at time period $t = 9$) is calculated as the rescheduling penalty incurred by changing the schedule at time period $t = 9$ compared to before rescheduling at time period $t = 9$. Note: although these events are generated as single rescheduling events, the events are in fact overlapping over time. Lastly, we provide the computation time by Time (s) for both approaches at each event.

Table 14: Simulation setting
$p_{max} = 80$, $i_{max} = 12$, $b_{max} = 4$, $l_{max} = 5$, $|\mathcal{L}_i|= 3$.

	Event	AC	TC	AP	TC	AC	AP
	Time period event generated	6	9	12	16	20	24
Cum. Location penalty	MIP	112	168	193	320	453	509
	H	112	187	200	278	299	366
Reschedule penalty	MIP	210	112	0	155	176	0
	H	198	117	15	140	134	54
Time (s)	MIP	6409	5678	>7200	6705	5967	>7200
	H	1115	1090	1693	1298	1214	2043

In this table we provide results for a simulation setting of the rescheduling problem for five instance over a time horizon of 24 time periods with six reschedule events. Each instance is generated by solving the initial planning problem using the heuristic approach with $p_{max} = 80$ people, $i_{max} = 12$ internships, $b_{max} = 4$ blocks, $l_{max} = 5$ locations $|\mathcal{L}_i|= 3$ locations per internship type, and thereafter the various rescheduling events are generated at the time periods provided below the given events. We provide

Based on Table 14 we conclude the MIP approach is significantly slower compared to the rescheduling approach, which is inline with the conclusions on single event rescheduling in Section 7.4.2. Second, we observe the MIP approach is outperformed in case of the rescheduling event where additional people are added to the traineeship program. This is also observed in Section 7.4.2 for single event rescheduling of additional people in Table 13. Lastly, we conclude based on the cumulative penalty the heuristic approach outperforms the MIP approach, which is mostly caused by the rescheduling event corresponding to additional people.

8 Conclusion

In this thesis we studied the collaborative traineeship planning problem. This problem consists of the construction of an initial traineeship planning and the rescheduling of an existing planning in case of disturbances. First, we described the problem in Section 2 and provided an overview of literature related to the problem in Section 3. The required data is described in Section 4. Thereafter, we formulated the initial planning problem as a MIP in Section 5. This MIP formulation was extended to formulate the rescheduling problem. Based on these formulations we proved both the initial planning and rescheduling problem are $\mathcal{NP}$-complete.

In this research we started with the following research question: *Can we design an algorithm that produces a collaborative traineeship planning within a reasonable amount of time while satisfying constraints on the traineeship program such that it is flexible in the trade-off of multiple objectives?*

We conclude the collaborative traineeship planning can be efficiently solved by two algorithms, which are specifically designed to solve either the initial planning or rescheduling variant of the problem in order to provide a solution within a reasonable amount of time. We list conclusions related to both variants of the problem in more detail below.

To solve the initial planning problem we first elaborated on variable fixing methods and additional valid inequalities to improve a MIP approach in Section 6.1.1. We analyzed the effect of these methods in Section 7.3.1 and concluded the MIP approach was performing best when variable fixing plus the first valid inequality (given by Equation (6.13)) was exploited. Next, we described a heuristic approach in Section 6.1.2 because of the performance requirement in terms of computation time. In this heuristic we considered a three-stage genetic algorithm, where the subproblem was formulated as a MIP in Section 6.1.2.6. We compare the heuristic in Section 7.3.2.1 with the MIP approach in terms of computation time. We concluded the heuristic approach outperforms the MIP approach, especially for practical applications where the MIP approach is not even able to find a feasible solution.

Two major components of this algorithm are analyzed in Section 7.3.2.2 and 7.3.2.3. We observed that an exact approach for the location subproblem was better than an exact approach where symmetry breaking constraints are added. This was caused by extra model building time for these constraints, which was larger than the reduction in solution time. Furthermore, we observed the greedy heuristic for the location problem was significantly faster than the exact approach while finding high-quality solutions, which supports the use of the greedy heuristic in the genetic algorithm. Lastly, we showed for the mathematical programming heuristic the objective values before and after this heuristic have an almost linear relationship, which supports the selection of schedules based on objective values.

For the rescheduling problem we again apply a MIP and heuristic approach. In the MIP approach we take advantage of variable fixing methods and valid inequalities. The heuristic approach consists of variable neighborhood search, where each neighborhood is evaluated by means of a tabu search procedure. Based on the results for single event rescheduling (see Section 7.4.2) and a simulation setting (see Section 7.4.3) we conclude the heuristic approach outperforms the MIP approach. Lastly, we evaluated the performance of the rescheduling algorithm in case this algorithm is applied to the initial planning problem. We concluded the initial planning heuristic outperforms the rescheduling heuristic for this particular setting.

For further research addressed to the CTPP we would like to extend the rescheduling algorithm, since from a practical point of view this problem shall be solved multiple times over a given time horizon in contrast to the initial planning problem. In our rescheduling heuristic we used only two neighborhoods, and hence we think there is some opportunity for improvement here. Furthermore, due to time limitations we have not been able to test various settings of the rescheduling penalty. That is, in further research this parameter shall be adjusted to match practical incentives more naturally.

Bibliography

Ağralı, S., Taşkın, Z. C., and Ünal, A. T. (2017). Employee scheduling in service industries with flexible employee availability and demand. *Omega*, 66:159–169.

Bard, J. F. and Purnomo, H. W. (2005). Preference scheduling for nurses using column generation. *European Journal of Operational Research*, 164(2):510–534.

Beliën, J. and Demeulemeester, E. (2004). Heuristic branch-and-price for building long term trainee schedules. *DTEW Research Report 0422*, pages 1–22.

Beliën, J. and Demeulemeester, E. (2006). Scheduling trainees at a hospital department using a branch-and-price approach. *European Journal of Operational Research*, 175(1):258–278.

Beliën, J. and Demeulemeester, E. (2007). On the trade-off between staff-decomposed and activity-decomposed column generation for a staff scheduling problem. *Annals of Operations Research*, 155(1):143–166.

Billionnet, A. (1999). Integer programming to schedule a hierarchical workforce with variable demands. *European Journal of Operational Research*, 114(1):105–114.

Brunner, J. O. and Edenharter, G. M. (2011). Long term staff scheduling of physicians with different experience levels in hospitals using column generation. *Health Care Management Science*, 14(2):189–202.

Brusco, M. J. and Jacobs, L. W. (1993). A simulated annealing approach to the solution of flexible labour scheduling problems. *Journal of the Operational Research Society*, 44(12):1191–1200.

Cordeau, J.-F., Stojković, G., Soumis, F., and Desrosiers, J. (2001). Benders decomposition for simultaneous aircraft routing and crew scheduling. *Transportation Science*, 35(4):375–388.

Dantzig, G. B. (1954). Letter to the editor -— a comment on Edie's "traffic delays at toll booths". *Journal of the Operations Research Society of America*, 2(3):339–341.

Dück, V., Ionescu, L., Kliewer, N., and Suhl, L. (2012). Increasing stability of crew and aircraft schedules. *Transportation research part C: emerging technologies*, 20(1):47–61.

Edie, L. C. (1954). Traffic delays at toll booths. *Journal of the Operations Research Society of America*, 2(2):107–138.

Ernst, A. T., Jiang, H., Krishnamoorthy, M., and Sier, D. (2004). Staff scheduling and rostering: A review of applications, methods and models. *European Journal of Operational Research*, 153(1):3–27.

Guo, J., Morrison, D. R., Jacobson, S. H., and Jokela, J. A. (2014). Complexity results for the basic residency scheduling problem. *Journal of Scheduling*, 17(3):211–223.

Haase, K., Desaulniers, G., and Desrosiers, J. (2001). Simultaneous vehicle and crew scheduling in urban mass transit systems. *Transportation Science*, 35(3):286–303.

Hanne, T., Dornberger, R., and Frey, L. (2009). Multi-objective and preference-based decision support for rail crew rostering. In *2009 IEEE Congress on Evolutionary Computation*, pages 990–996. IEEE.

Hoffman, K. L. and Padberg, M. (1993). Solving airline crew scheduling problems by branch-and-cut. *Management Science*, 39(6):657–682.

Huisman, D. and Wagelmans, A. P. (2006). A solution approach for dynamic vehicle and crew scheduling. *European Journal of Operational Research*, 172(2):453–471.

Ionescu, L. and Kliewer, N. (2011). Increasing flexibility of airline crew schedules. *Procedia-Social and Behavioral Sciences*, 20:1019–1028.

Juang, Y.-S., Lin, S.-S., and Kao, H.-P. (2007). An adaptive scheduling system with genetic algorithms for arranging employee training programs. *Expert Systems with Applications*, 33(3):642–651.

Karp, R. M. (1972). Reducibility among combinatorial problems. In *Complexity of computer computations*, pages 85–103. Springer.

Kohl, N. and Karisch, S. E. (2004). Airline crew rostering: Problem types, modeling, and optimization. *Annals of Operations Research*, 127(1-4):223–257.

Maenhout, B. and Vanhoucke, M. (2011). An evolutionary approach for the nurse rerostering problem. *Computers & Operations Research*, 38(10):1400–1411.

Moz, M. and Pato, M. V. (2004). Solving the problem of rerostering nurse schedules with hard constraints: New multicommodity flow models. *Annals of Operations Research*, 128(1-4):179–197.

Moz, M. and Pato, M. V. (2007). A genetic algorithm approach to a nurse rerostering problem. *Computers & Operations Research*, 34(3):667–691.

Parr, D. and Thompson, J. M. (2007). Solving the multi-objective nurse scheduling problem with a weighted cost function. *Annals of Operations Research*, 155(1):279–288.

Rasmussen, M. S., Justesen, T., Dohn, A., and Larsen, J. (2012). The home care crew scheduling problem: Preference-based visit clustering and temporal dependencies. *European Journal of Operational Research*, 219(3):598–610.

Stolletz, R. and Brunner, J. O. (2012). Fair optimization of fortnightly physician schedules with flexible shifts. *European Journal of Operational Research*, 219(3):622–629.

Van den Bergh, J., Beliën, J., De Bruecker, P., Demeulemeester, E., and De Boeck, L. (2013). Personnel scheduling: A literature review. *European Journal of Operational Research*, 226(3):367–385.

Veelenturf, L. P., Potthoff, D., Huisman, D., Kroon, L. G., Maróti, G., and Wagelmans, A. P. (2014). A quasi-robust optimization approach for crew rescheduling. *Transportation Science*, 50(1):204–215.

Yen, J. W. and Birge, J. R. (2006). A stochastic programming approach to the airline crew scheduling problem. *Transportation Science*, 40(1):3–14.

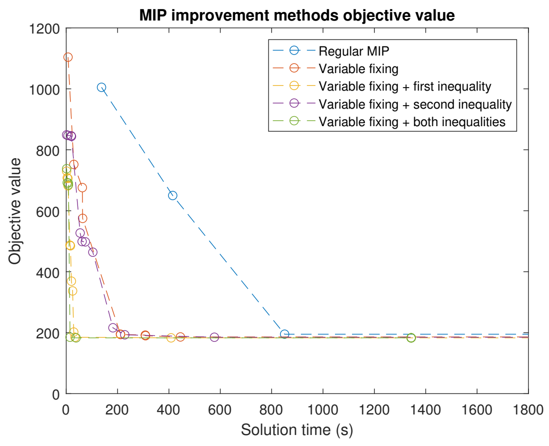
Appendix A

Table 15: Initial planning problem
$i_{max} = 4$, $b_{max} = 2$, $l_{max} = 2$, $|\mathcal{L}_i|= 2$

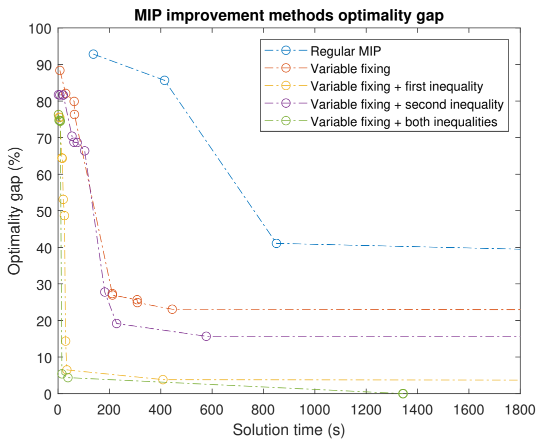
p_{max}	Gap MIP (%)	Gap H (%)
5	0.00	0.00
6	0.00	0.00
7	0.00	0.00
8	0.00	0.00
9	0.00	0.00
10	0.00	0.00
11	0.00	1.56
12	0.00	0.00
13	0.00	1.04
14	0.00	0.00
15	0.00	0.00
16	0.00	0.00
17	0.00	0.00
18	0.00	0.00
19	0.00	0.00
20	0.00	0.00
21	0.00	0.00
22	0.00	0.00
23	0.00	0.00
24	0.72	0.73
25	0.00	0.00
26	0.00	0.00
27	0.00	0.00
28	0.00	0.45
29	0.00	1.10
30	0.00	0.00
31	0.00	0.39
32	0.00	0.81
33	0.00	0.00
34	0.00	0.00
35	1.12	1.13
36	0.00	0.00
37	0.00	0.00
38	0.00	0.00
39	0.51	0.51
40	1.22	1.88
41	0.00	0.00
42	0.00	0.00
43	0.00	0.00
44	0.00	0.00

In this table we provide results for the initial planning problem for instances with $i_{max} = 4$ internships, $b_{max} = 2$ blocks, $l_{max} = 2$ locations and $|\mathcal{L}_i|= 2$ locations per internship. The number of people is given by p_{max} and corresponds to one instance. The optimality gap for the MIP and heuristic approach are given by Gap MIP (%) and Gap H (%), respectively. The solution time for the MIP approach was restricted to a maximum of 1800 seconds per instance.

Appendix B

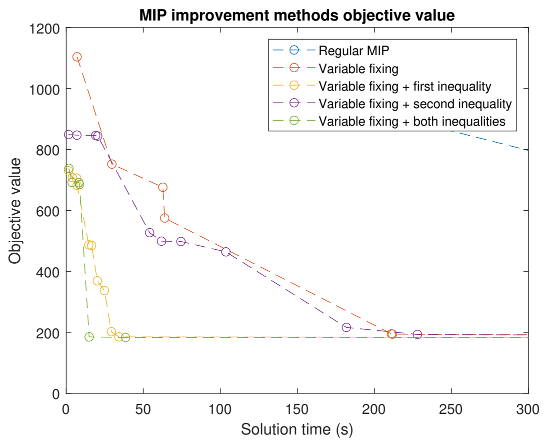


(a) Objective value.

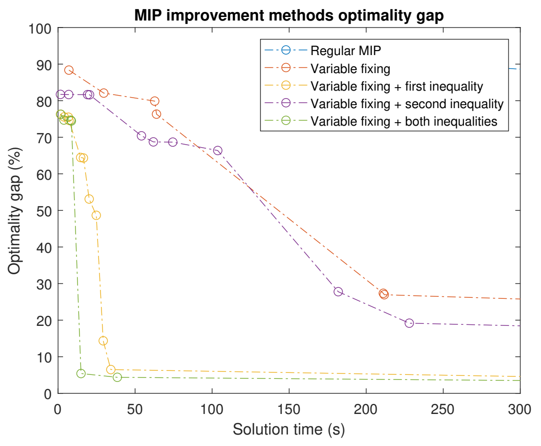


(b) Optimality gap.

Figure 9: Effect of MIP improvement methods.

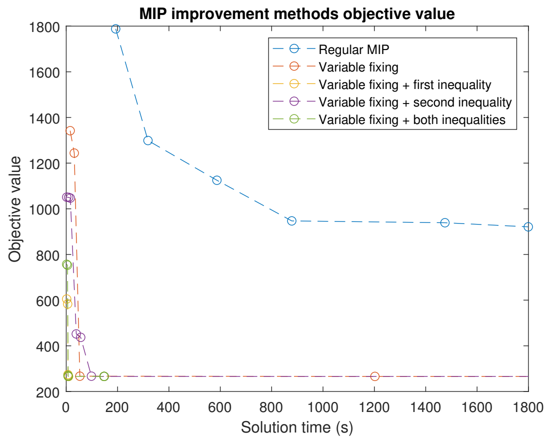


(a) Objective value.

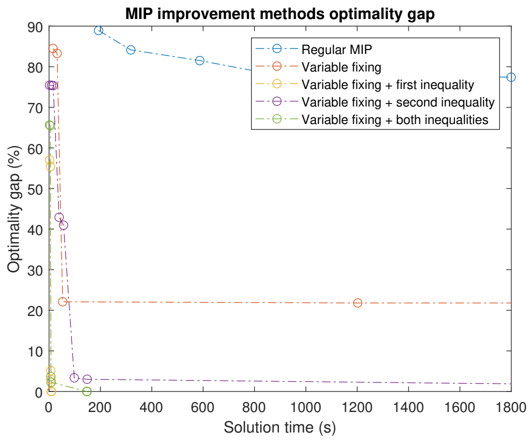


(b) Optimality gap.

Figure 10: Effect of MIP improvement methods detailed.

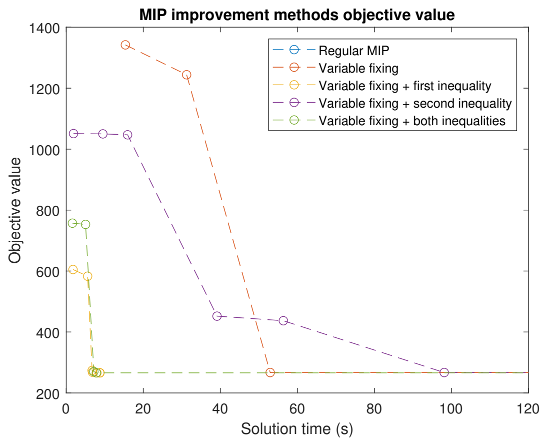


(a) Objective value.

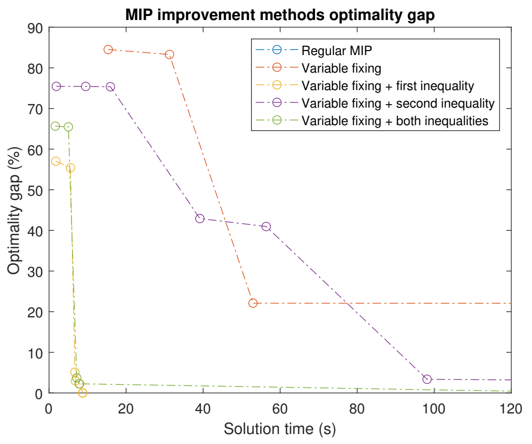


(b) Optimality gap.

Figure 11: Effect of MIP improvement methods.



(a) Objective value.



(b) Optimality gap.

Figure 12: Effect of MIP improvement methods detailed.

Appendix C

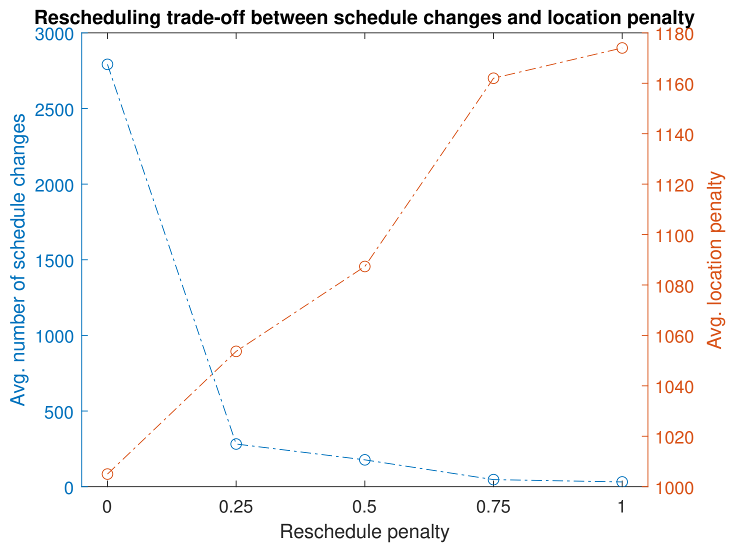


Figure 13: Rescheduling trade-off instances with $p_{max} = 80$ people.