Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Economics and Management
Dept. of Industrial Economics and Technology Management

Håkon Grov
Even Kallevik
Sondre Nærland

Optimizing Real-Life Personnel Scheduling Problems through Shift Design and a Parallel Adaptive Large Neighborhood Search

Master's thesis in Managerial Economics and Operation Research
Supervisor: Anders Nordby Gullhav
July 2020

NTNU
Norwegian University of Science and Technology

Preface

This master's thesis concludes our Master of Science at the Norwegian University of Science and Technology (NTNU), with specialization in Applied Economics and Optimisation at the Department of Industrial Economics and Technology Management. The thesis is written in collaboration with Visma Optimizations Technologies and Medvind, and is a continuation of the preliminary work in Grov et al. (2019).

First and foremost, we want to express our sincere gratitude to our supervisor, Associate Professor Anders Nordby Gullhav, from the Department of Industrial Economics and Technology Management at NTNU, for his valuable guidance and constructive feedback.

We also want to thank Jacob Nyman and Jacub Jonik from Visma Optimization Technologies for their help on feedback and providing necessary information, and for operating as the intermediary to Medvind Workforce Management. We especially want to thank Jacob Nyman, who has supported us in particular and guided us throughout the project.

Håkon Grov, Even Kallevik and Sondre Nærland
Trondheim, July 2020

Abstract

Service industries are faced with the challenging task of efficiently satisfying a varying demand with the right supply of qualified personnel. To schedule personnel in order to sustain a desirable service level is, therefore, an important concern. Additionally, recent studies have revealed the importance of employee satisfaction and perceived fairness of schedules, as this tends to increased work performance. This aspect increases the complexity of creating high-quality work schedules.

Medvind, a provider of workforce management systems, are currently developing a scheduling tool that aims to generate optimized work schedules. They have a broad base of customers, which require the scheduling model to be general in order for it to apply to the variety of service industries represented. In this master's thesis, we address the problem faced by Medvind, defined as the Medvind Scheduling Problem (MSP). Our purpose is to generate high-quality schedules, which implies that demand is covered and that the fairness of the employees is maximized.

In order to achieve high-quality schedules, this thesis address two important aspects of personnel scheduling. First, we address the shift design problem, concerned with the generation of favourable shifts for the rostering problem. Secondly, we address the rostering problem, concerned with allocating staff to the generated shifts in an optimal way. In this thesis, we propose a Shift Design Heuristic (SDH), aiming to create shifts tailored to the faced demand, while allowing flexibility to satisfy the fairness aspects considered in the MSP. For the rostering problem, a Mixed Integer Programming (MIP) model has been proposed with the ability to find the optimal schedules. As the performance of exact solvers tends to diminish significantly with higher complexity, a Parallel Adaptive Large Neighborhood Search (PALNS) is also implemented in order to better cope with rostering problems of increased complexity.

When evaluating the SDH on nine real-life problems provided by Medvind, the SDH generates shifts that clearly outperform manually created shifts in terms of performance and efficiency. The generated shifts are also compared to ideal shifts generated by an implicit scheduling model that was developed in the preliminary work of this thesis. This implicit scheduling model generates shifts implicitly, allowing for more or less unlimited flexibility but at the expense of increased complexity in obtaining scheduling solution. The comparison is thus performed on simplified test problems. Even though the implicitly generated

shifts facilities the best schedules, the SDH is able to design shifts similar to the implicit shifts and significantly outperform the implicit model when it comes to practical applicability.

Comparing the MIP model to the PALNS shows an advantage for the MIP model on the less complex problem instances. However, as the size and complexity increases, the performance of the MIP model decreases, resulting in no feasible solution for three of the nine provided test instances even with a time limit of five hours. The PALNS, on the other hand, is able to achieve satisfactory solutions for all real-life problems provided by Medvind. Although the obtained objective values are lower for the PALNS compared to the MIP model, these differences are not significant in real-life applications. Furthermore, the PALNS is considerably faster to obtain solutions for more complex problems. Additionally, the PALNS seems to scale well with both the number of employees and the length of the planning horizon. This does not apply to the MIP model. The ability to efficiently create high-quality schedules for problem instances of varying complexity indicates that the PALNS is the preferred solution approach for the MSP.

The combination of the SDH and the PALNS presented in this thesis greatly succeed in generating high-quality schedules within the desired time limit, that both satisfy demand and facilitate a high degree of fairness. The PALNS is able to obtain good schedules for all real problems provided by Medvind, underlining the ability of the PALNS to handle general scheduling problems. Additionally, the developed SDH is able to generate favourable shifts tailored to each problem in a very efficient manner. This contribution is assumed to have a significant practical value.

Sammendrag

Tjenesteytende industrier står overfor den utfordrende oppgaven med å effektivt tilfredsstille en varierende etterspørsel med riktig tilførsel av kvalifisert personell. Å allokere personell til arbeidsskift, slik at et ønskelig tjenestenivå opprettholdes, er en viktig og krevende oppgave for mange selskaper og organisasjoner. Fordelingen av arbeidsskift mellom de ansatte er det vi heretter referer til som arbeidstimelister. I tillegg har nyere studier avdekket viktigheten av ansattes tilfredsshet og opplevd rettferdighet, da disse aspektene kan bidra til å øke arbeidsytelsen. Å ihensynta slike aspekter øker kompleksiteten ved lage tilfredstillende arbeidstimelister ytterligere.

Medvind, en leverandør av systemer for arbeidsledelse, utvikler et planleggingsverktøy som skal kunne generere optimaliserte arbeidstimelister. De har et bredt spekter av ulike kunder, noe som krever at verktøyet for generering av arbeidstimelister er tilstrekkelig generelt, slik at den kan anvendes i de mange industriene som er representert i Medvinds kundebase. I denne masteroppgaven adresserer vi problemet Medvind står overfor, heretter definert som MSP. Vår målsetning er å utvikle en løsningsmetode som genererer arbeidstimelister av høy kvalitet, noe som innebærer at arbeidstimelistene dekker etterspørsel og at rettferdighet maksimeres blant de ansatte.

For å oppnå arbeidstimelister av høy kvalitet, adresserer denne oppgaven to viktige sider ved personalplanlegging; skiftdesign-problemet og allokeringsproblemet. Skiftdesign-problemet omhandler utfordringene med å designe hensiktsmessige arbeidsskift som de ansatte kan tildeles, mens allokeringsproblemet omhandler utfordringene med å allokere de ansatte til de genererte skiftene. I denne masteroppgaven foreslår vi en heuristisk metode for å designe skift (Shift Design Heuristic - SDH), hvor skiftene er tilpasset etterspørselen og samtidig sørger for å fasiliterere fleksibilitet for å bedre imøtekomme rettferdighetsaspektene som betraktes i MSP. For allokeringsproblemet foreslår vi to løsningsmetoder; en blandet heltallsløser (Mixed Integer Programming model - MIP-modell) og et parallelt adaptivt stort nabolagssøk (Parallel Adaptive Large Neighbourhood Search - PALNS). MIP-modellen er en eksakt løser som skal sørge for å generere optimale arbeidslisteplaner. En utfordring med eksakte løsere er imidlertid at ytelsen ofte reduseres vesentlig når kompleksiteten ved allokeringsproblemene øker. Derfor foreslås PALNSen, som har til hensikt å generere løsninger raskere og skalere bedre ved økende komplekistet.

Den foreslåtte SDHen testes på ni reelle probleminstanser gitt av Medvind. Resultatene fre disse testene viser at SDHen genererer skift som klart utkonkurrerer manuelt genererte skift når det kommer til ytelse og effektivitet. Skiftene som genereres av SDHen sammenlignes også med ideelle skift generert av en implisitt arbeidstimelistemodell utviklet i forarbeidet til denne masteroppgaven. Denne implisitte modellen tar ikke i bruk utviklede skift, men allokerer ansatte til starttidspunkter med tilhørende varigheter. Dette innebærer tilnærmet ubegrenset fleksibilitet, noe som legger til rette for ideelle skift. Den betydelige fleksibiliteten medfører imidlertid høy kompleksitet, noe som gjør det krevende for modellen å lage arbeidstimelister. Sammenligningen mellom SDHen og de implisitte skiftene gjøres derfor på forenklede probleminstanser. Selv om de implisitte skiftene muliggjør de beste arbeidstimelistene, er SDHen i stand til å designe skift som ligner de implisitte, og SDHen overgår i tillegg klart den implisitte modellen når det kommer til praktisk anvendbarhet.

Sammenligningen av MIP-modellen og PALNSen viser at MIP-modellen yter bedre på probleminstanser som kan betraktes som mindre komplekse. Når størrelsen og kompleksiteten på problemene øker, reduseres MIP-modellens ytelse. Dette gjør seg gjeldende ved at MIP-modellen ikke evner å generere en mulig løsning for tre av de ni probleminstansene gitt av Medvind, innen tidsbegrensningen på fem timer. PALNSen er derimot i stand til å oppnå tilfredsstillende løsninger for alle de reelle problemene, selv innenfor 15 minutter. Selv om den oppnådde objektivverdien er lavere for PALNSen, sammenlignet med MIP-modellen, er denne forskjellen i praksis neglisjerbar. PALNSen evner også å generere løsninger vesentlig kjappere enn MIP-modellen for mer komplekse problemer, og skalerer i tillegg godt når antall ansatte eller planleggingshorisonten økes for de reelle problemene. Dette gjelder ikke for MIP-modellen. Evnen til å effektivt generere arbeidslisteplaner av høy kvalitet for probleminstanser av varierende kompleksitet, indikerer at PALNSen er den foretrukne løsningsmetoden for MSP.

Kombinasjonen av de utviklede heuristikkene, SDH og PALNS, evner i stor grad å generere arbeidstimelister av høy kvalitet innenfor ønsket tidsbegrensning. Disse arbeidstimelistene tilfredsstiller etterspørselen og legger til rette for en høy grad av rettferdighet. PALNSen er i stand til å generere arbeidstimelister for alle de varierte og reelle problemene gitt av Medvind, som understreker PALNSens evne til å håndtere generelle planleggingsproblemer. I tillegg er den utviklede SDHen i stand til å generere gunstige skift skreddersydd for hvert problem på en veldig effektiv måte. Dette bidraget antas å ha en betydelig praktisk verdi.

Table of Contents

List of Tables

List of Figures

List of Algorithms

List of Abbreviations

Medvind WFM	Medvind Workforce Management
MIP	Mixed Integer Programming
MIP model	Mixed Integer Programming model
IMP-MIP model	Mixed Integer Programming model with implicitly generated shifts
ALNS	Adaptive Large Neighbourhood Search
PALNS	Parallel Adaptive Large Neighbourhood Search
MSP	Medvind Scheduling Problem
SDH	Shift Design Heuristic
SR extension	Shift Reduction extension
SDH-SR	Shift Design Heuristic with Shift Reduction extension
CSP	Constraint Satisfaction Problem

Chapter 1

Introduction

The service sector constitutes a significant part of the modern economy and continues to grow. Within this sector, the satisfaction of customer demand is of high importance, a demand that can significantly vary in both characteristics and magnitude, and which cannot be backlogged. As the service sector tends to be labor-intensive, personnel are commonly the most expensive resource. Organizations are, therefore, striving to meet varying demands while continually looking for opportunities to operate efficiently. The efficient utilization of labor has become increasingly important, and Ağralı et al. (2017) states that it is the most important concern in service industries today.

Personnel scheduling problems have been widely studied in the last decades. The research attention is motivated by economic considerations, as well as the complexity of matching demand with a qualified workforce. In many industries, it can be tedious and difficult to schedule personnel in order to satisfy demand, while at the same time adhering to laws and regulations stated by governmental entities and trade unions. The outcome is often constrained and highly complex optimization problems, considered one of the most challenging tasks a manager can face (De Bruecker et al., 2015). In addition, employee satisfaction and perceived fairness of the schedules have received increased attention in recent years. This increases the complexity of scheduling further. Studies have indicated that accounting for employee preferences, avoiding undesired shift patterns and making fair schedules could lead to increased work performance (Bard and Purnomo, 2005; Smet et al., 2012). As a result, many industries face two main questions that need to be addressed: how to organize a workforce efficiently to cover demand, and how to create schedules that increase employee satisfaction and the perceived fairness. As stated in Lapègue et al. (2013), it is often difficult to make these goals compatible.

Medvind, part of the Visma Group, is a provider of a workforce management system, aiming to support workforce managers with a variety of planning tasks. Medvind is currently expanding its product portfolio of planning tools by developing a scheduling model that aims to create an optimized work schedule based on input parameters from workforce managers. The scheduling solutions need to be general to make it applicable to Medvind's wide range of customers, representing a variety of different industries.

The purpose of this master's thesis is to develop a solution method that solves the scheduling problem faced by Medvind. This problem is referred to as the Medvind Scheduling Problem and hereby donated the MSP. The solution method should be able to handle general and varied personnel scheduling problems by generating work schedules that satisfy all requirements stated by Medvind. The overall goal is to allocated employees to shifts in order to meet a predefined demand as close as possible, and the solution method should aim to maximize the perceived fairness of the schedules.

To facilitate the best possible work schedules, generated within a practical time frame, this master's thesis explore two aspects within personnel scheduling in particular:

(i) The effects of heuristically generated shifts for scheduling problems.

(ii) How to design a solution method capable of achieving high-quality schedules for the MSP and its wide range of problem sizes and complexities.

Shifts used in personnel scheduling are often created manually, which is a time-consuming process resulting in shifts that do not necessarily facilitate favorable schedules. In this thesis, we explore the effects of automatically generating shifts tailored to the problem instances of the MSP. A shift design heuristic is proposed, hereby denoted the SDH, intending to generate shifts that facilitates ideal demand coverage, and that includes flexibility in order to provide fair schedules. As the MSP is a general problem, the solution approach is required to handle a wide range of problem instances with different degrees of complexity. In order to address the second area of exploration, we investigate two different solution methods for allocating employees to shifts. First, we propose a Mixed Integer Programming model, before an alternative heuristic model applying a Parallel Adaptive Large Neighbourhood Search is proposed. These two models are hereby denoted the MIP-Model and the PALNS, respectively.

This thesis is a continuation of a preliminary work, where the MSP was solved with an implicit scheduling model, that allocated employees to a starting time with an associated duration. The shifts were thus implicitly given. This scheduling model is hereby denoted the IMP-MIP model. The IMP-MIP model allowed for more or less unlimited flexibility but was a highly complex model to solve, as the number of possible shifts quickly became unmanageable. The model provided strong results on small artificial problems but struggled with the size and complexity of real-life problems. The preliminary work serves as a backdrop for this master's thesis and is of importance in some of the modeling choices made.

The main contributions of this thesis are the developed SDH and PALNS, which, combined, efficiently solve the MSP. The SDH is a novel heuristic for shift design, aiming to utilize demand information in order to generate shifts that facilitate high-quality schedules. The shift design approaches, as applied in the SDH is, to our knowledge, not covered in the existing literature. The proposed SDH succeeds in providing shifts that facilitate high-quality schedules with high efficiency, making it a superior approach compared to the common use of manually created shifts. The proposed PALNS is able to solve the

MSP within a highly restricted time frame, considerably outperforming both a conventional ALNS and the implemented MIP model. Our academic contribution is to provide insight into parallelization of heuristics, as well as the applicability of the ALNS framework for personnel scheduling problems. The PALNS provide promising solutions, and yields several benefits over the conventional ALNS. Furthermore, by successfully applying an implementation of the ALNS framework to the MSP we provide interesting results on the applicability of the heuristic and present new possibilities within the field of personnel scheduling.

A further elaboration on the background for this master's thesis is given in Chapter 2, providing insight into Medvind and the MSP. In Chapter 3, relevant literature is presented, and the problem is positioned within a literary context. Chapter 4 presents the scope and a description of the MSP. In Chapter 5, the SDH is presented and further motivated. The proposed Mixed-Integer Programming model is presented and discussed in Chapter 6, while Chapter 7 includes an elaboration on the developed PALNS. Chapter 8 presents a number of real-life problem instances used for testing, in addition to all parameters used in the solution approaches. The results of the computational study are discussed and evaluated in Chapter 9. The thesis ends with Chapter 10, summarizing and concluding on the contribution of this master's thesis along with a presentation of possible areas for further research.

Chapter 2

Background

The purpose of this chapter is to present background information relevant to the thesis. Section 2.1 introduces terminology that is important for the understanding of the subsequent chapters. Section 2.2 gives a motivation for the area of personnel scheduling in a wider context, before Medvind and their desire to create a general workforce management system is introduced in Section 2.3. Finally, Section 2.4 presents the common characteristics of personnel scheduling problems present in the industries that the Medvind WFM system aims to target.

2.1 Terminology

This section presents the most useful terms and definitions used in the thesis, aiming to provide a better understanding of the remaining sections and chapters.

- *Shift pattern* - A specific combination of subsequent shifts.
- *Staffing* - The process of employing a workforce to cover specific job functions.
- *Staff scheduling* - The process of planning turns of duty for individuals or groups in an organization.
- *Schedule* - The planned turns of duty for all personnel is called a schedule.
- *Scheduling rules* - Rules, typically defined by governmental work regulations, trade unions, and preferred practices, that states which schedules are legal or not.
- *Scheduling flexibility* - The ability to have multiple allocation options of shifts.
- *Scheduling manager* - Person or group responsible for staff scheduling.
- *Planning period/horizon* - The period for which the schedule is created.
- *Contracted hours* - Contracted number of working hours the employee is to work each week.

- *Bartering* - The process of exchanging shifts among employees after a schedule is generated.
- *Fairness aspects* - Aspects that affect the perceived fairness of a schedule.
- *Time step* - Time resolution of real-life problem instances. Usually 15, 30 or 60 minutes.
- *Blocked hours* - Periods where an employee is unavailable for work.
- *Partial weekends* - An undesired shift pattern where an employee works either Saturday or Sunday, but not both.
- *Calendar day* - A day starting at 00:00 and ending at 24:00.
- *Calendar week* - A week starting at Monday 00:00 and ending the following Sunday at 24:00.
- *Employee-specific day* - A 24-hour interval with a possible offset from calendar days. If an employee-specific day starts at 4 a.m, the employee-specific day thus regards the following 24 hours.
- *Daily rest* - The required amount of continuous rest for en employee within an employee-specific day.
- *Weekly rest* - The required amount of continuous rest for an employee within a calendar week.
- *Workforce Management System* - System that helps a business manage staff scheduling.

2.2 Personnel Scheduling

As indicated in Chapter 1, personnel scheduling has been a central theme in workforce management for decades. The backdrop for the attention directed at personnel scheduling is historically justified by cost considerations and because these problems have been, and still are, highly complex to solve. Personnel scheduling problems are common and relevant in a number of industries and thus impact on a large part of society. With an economy and a society that is turning towards increased concentration and demand for service-related businesses, personnel scheduling problems are more relevant than ever before. As demand and customer satisfaction are increasingly dependent on the executing personnel, and with personnel cost constituting the primary source of expenses for the majority of industries, effective utilization of human resources has in earnest established its importance.

Today, personnel scheduling is mostly performed manually. When forecasts of demand are completed and the proper staff is selected, managers or appointed employees start the process of scheduling the staff in order to satisfy the demand. The scheduling includes both the selection of shifts to be used throughout the planning period and the allocation

of personnel to these shifts in a satisfactory way. Both of these processes are tedious and complex to solve, and in most cases, it is hard even to find a feasible solution that complies with all laws and regulations. The outcome is often poor or insufficient schedules, causing over- or understaffing of personnel, violations of work regulations, contractual agreements and employee preferences. These inefficiencies have a negative influence on cost aspects, staff satisfaction and the obtained service level.

Personnel scheduling problems are highly complex optimization problems, as there are several decisions to be made and many considerations and requirements that must be taken into account. In addition, there are often large variations in the nature of the problems between industries, making it challenging to establish useful frameworks to encounter the general scheduling problem. As a result, there are today very few practical models that can solve real-life personnel scheduling problems.

In later years, the process of generating schedules that are fair to the employees has received increased attention. Research indicates that schedules that account for employee preferences, and that are perceived as fair by the staff, facilitates increased work performance and a healthier work environment (Bard and Purnomo, 2005; Smet et al., 2012). By taking these aspects into account, the complexity of the personnel scheduling problems is increased further. This makes it hard to exploit the effects of fair schedules. Industries have, for a long time, yearned for a solution that is able to cope with the complexity of personnel scheduling problems and that enables efficient workforce management. With advances in both research and computer performance there exists an enormous potential for improvements on today's situation, and better solution approaches for personnel scheduling problems would create tremendous value to a number of industries.

2.3 Medvind WFM

Medvind WFM is a workforce management system that emerged in 1995, to be used in the Swedish market. Medvind is today part of the Visma Group, and its WFM system is currently used by approximately one hundred corporate customers internationally in both private and governmental sectors, ranging from nursing homes to hotels and pharmacies. Being a comprehensive workforce management system, Medvind WFM can handle a large number of strategic activities within human resource management and is also integrated into all of the human resource management systems created by Visma. These include Agda, HR-plus and Personec.

Medvind WFM is created as a general-purpose workforce management system with the ability to adapt to the different workforce management models, used in various industries. There are many challenges when aiming to create a workforce management system applicable to multiple industries. The system has to be able to take into account legislation, regulation, and local "game rules" specific to the industry at hand. Besides, the industry-specific demand characteristics might differ greatly from one another. Furthermore, making sure assigned employees have the correct certifications and training is highly important, and differs significantly between industries. To overcome these challenges, Medwind

WFM is developing a general workforce management system that is based on the characteristics of workforce management common to all industries. The system is then configured to fit the customers and industries at hand, accounting for the different individualities. The configuration relies on accurate input data from the customers, which includes legislation and regulations specific to that business, a forecast of the future demand, all relevant information regarding the workforce, as well as preferences and other relevant aspects provided by the employees. The advantage of the Medvind WFM system is that it constitutes a service that can create value for a wide range of customers and businesses. Medvind is currently aiming to extend its product portfolio by adding a workforce scheduling system that is able to allocate employees to shifts in order to meet a specific demand.

2.4 The General Scheduling Problem

The Medvind WFM system is not created with one particular industry in mind. It is developed to be able to handle multiple industries and to ensure that the need for industry-specific adaptions and adjustments are kept to a minimum. A visualization of the overall planning process of the Medvind WFM system is presented in Figure 2.1 and consists of three phases; the planning phase, the prerequisites phase and the scheduling phase. In the planning phase, the planning is on a strategic level. The first objective is to forecast the expected demand, like numbers of kids in the kindergarten or number of surgeries to be performed. The second objective is then to plan the required resources to satisfy the expected demand, like staffing level, needed competencies and which tasks to be performed. In the prerequisites phase, specifications like laws and regulations, local game rules and employee stats and preferences are stated. Note that the *best practice* is also defined in this phase, indicating what constitutes an acceptable schedule. The final phase is the scheduling phase, where an actual schedule is created. The proposed schedule is evaluated, and if the desired result is not obtained, adjustment on the ”best practice” or input data is performed. The MSP concerns the area of scheduling proposals in the scheduling phase.

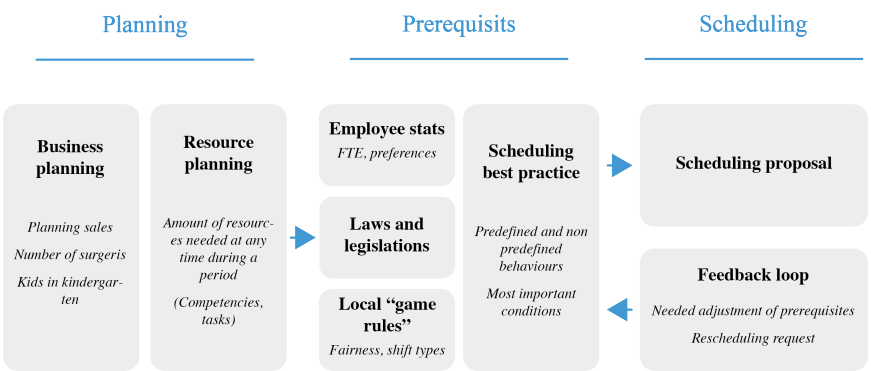

Figure 2.1: The overall planning process of the Medvind WFM system

To be able to provide a general WFM system, Medvind has to build the system around the scheduling characteristics common to all relevant industries. In the remainder of this section, the most prominent characteristics concerning personnel scheduling will be presented in a thematic order.

2.4.1 Demand

The demand an organization faces has a crucial influence on the operations of the organizations. The nature of demand varies greatly among different industries. However, one common characteristic is that demand often is related to specific requirements, like required competencies, amount of workers and the time step of the demand. To predict demand is challenging, but through the identification of demand characteristics, use of historical data and estimates of future parameters, it is possible to perform some forecasting and modeling of the demand. This forecast is then used to design shifts or perform appropriate measures regarding staffing, and it is therefore desirable that the forecasted demand is as realistic as possible. If the forecast is inaccurate, it could lead to either over- or understaffing, two common sources of inefficiencies. If there are allocated too few workers, patients might get a lower level of service, and it will put more pressure on the employees working. On the other hand, allocating to many employees will be inefficient and costly. The forecasting of demand is thus an essential part of being able to allocate personnel to shifts in a satisfactory and efficient manner. Medvind WFM leaves the customer in charge of the demand forecast by letting the customer define a minimum, ideal, and maximum staffing level and the set of competencies needed to satisfy the demand.

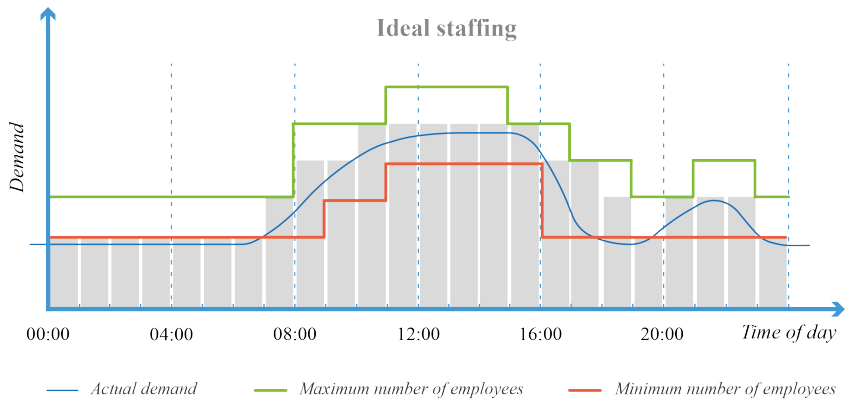

Figure 2.2: Example of forecasted and discretized demand curve

Figure 2.2 illustrates an example of how a forecasted demand curve might look in cases with demand spanning 24 hours. Note that the gray bars represent ideal demand. Note also that a demand curve is often discretized for practical reasons. As a single curve typically

represents the need for one particular set of competencies, there might be several demand curves throughout a day that together constitutes the aggregated demand, which needs to be met.

2.4.2 Shifts

Shifts exist in many different shapes. A shift is often designed to meet a specified demand or a set of demands, where the demand often has a particular duration and requirements regarding specific competencies and the required number of employees. When a sales clerk assists a customer in a grocery store, an off-shore worker is stationed on a platform, or a nurse is working throughout the night, they are all allocated to some sort of shift. Even though all of these shifts are different in its nature, they all have in common that they are defined for a fixed duration of time and with specific requirements, often regarding required competencies and the number of employees.

Some shifts are often considered less attractive to work. Weekend shifts, shifts that fall on holidays and in some instances shifts that involve working nights are usually less popular. Also, certain combinations of shifts might be unpopular as they lead to higher strain on the employees. Such combinations are considered unwanted shift patterns, and could also be based on personal preferences. An example of an unwanted shift pattern is typically partial weekends, where an employee only works either Saturday or Sunday. To create fair schedules, the less attractive shifts need to be accounted for. In some cases, the assignment of particular shift and shift patterns are also prohibited by the working environment act.

2.4.3 Shift Design

As mentioned, shifts are typically designed in order to cover demand. In many industries, especially those including demand that is spanning 24 hours like hospitals and nursing homes, shifts typically occur in the form of day, evening and night shifts. With a demand spanning 24 hours, these shifts are designed to enclose the entire demand period, as shown in Figure 2.3.

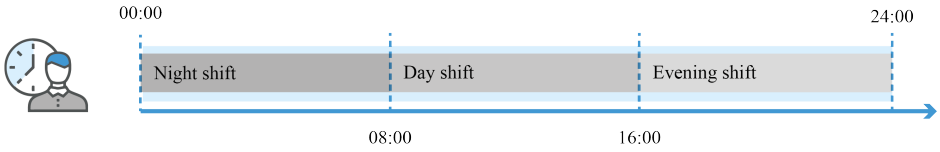

Figure 2.3: Classical example on day, evening and night shift.

There may exist several different variations of the day, evening or night shifts, with varying start time and duration. However, they can often be defined within one of the aforementioned shift scopes. To ensure flexibility, shifts are additionally often designed with some

overlap to ensure a closer workforce fit to a demand varying throughout the day and to prevent problems, and facilitate the transmission of information, between shift exchanges. It is, however, a common practice to reduce the number of different shifts and to design certain base shifts that apply to most of the days in the planning period. Today, the Medvinds WFM system lets the customer design its desired set of shifts.

2.4.4 Employees

An organization consists of employees, the resource that typically performs the work. To cover demand, employees are commonly allocated to shifts. The characteristics and amount of employees constituting the workforce varies significantly between industries and organizations. Some common characteristics for employees are, however, still identifiable. Each of these common characteristics will be described in further detail below.

Competencies

Employees possess different competencies, often based on educational level, training or experience. The competencies of an employee determine which tasks the employee could be assigned to, and thus which demand that can be covered.

Contractual Agreements

Common, in more or less every industry, is that an employee is subject to an employment contract. Such a contract will typically state the contracted hours of the employment, i.e., how many hours the employee should work during the planning period and which periods the employee is to work. An employee should be scheduled to work their contracted hours, even though small deviations are allowed. An employee is often restricted to not work more than the contracted hours, or at least not more than a certain amount of overtime, due to limitations in governmental laws and work regulations. Employees are typically paid based on their contracted hours and not their actual working hours. The deviation is therefore considered an unnecessary cost, and should thus be avoided.

In addition, contractual agreements could entail employee availability. Before the beginning of the planning period, employees might know of specific periods or shifts they are unable to work. This might be because of appointments, personal inclinations or other reasons specific to the individual employee. Such periods are considered blocked hours and are typically specified in the individual employment contracts.

Rest

Employees need rest between shifts, and the requirements related to rest is often based on governmental laws or work regulations through trade union agreements. The required rest will depend on the nature of the work, and thus varies a lot among industries. Common features are, however, that there should be a certain amount of continuous daily and weekly rest hours, that only one shift can be worked each day and that there is a certain off-period between two subsequent shifts.

Personal Preferences

To facilitate shift allocations that are favorable for the employees, it is increasingly common that employees are allowed to submit preferences regarding desired work hours, off periods or certain shift patterns. Schedules that are preferred by the employees are often easier to implement and result in less bartering and adjustments of the schedule in afterhand. Research also indicates that it leads to better health, less absenteeism and increased work performance (Bard and Purnomo, 2005; Smet et al., 2012).

Chapter 3

Literature Review

In this chapter, relevant literature to the MSP and personnel scheduling in general is presented. In Section 3.1, the MSP is positioned within the wide scope of relevant literature. Based on the positioning, Section 3.2 present the literature we consider most relevant to this thesis. The scope is then narrowed down in Section 3.3 and Section 3.4, where we describe the most common shift design strategies and personnel scheduling strategies, respectively. The key concepts of personnel scheduling are presented in Section 3.5 , while Section 3.6 elaborates on the most common optimization approaches relevant to the MSP. Finally, the literature review is concluded and summarized in Section 3.7.

3.1 Positioning in the Literature

How to efficiently allocate resources in an organization with respect to costs, scheduling quality and other aspects, is a question of high relevance to many industries. This theme has been an insatiable topic for decades and is broadly covered in the literature. In this section, the MSP is positioned within the existing literature. We start broad by introducing relevant literature covering planning and control before narrowing the scope by presenting relevant literature on the classification of personnel scheduling.

3.1.1 Planning and Control

An essential part of operational management is planning and control. In the literature, many frameworks have been proposed for this topic, aiming to structure the processes and to enable identification of various types of managerial problems. As presented in Zijm (2000), most of the frameworks are typically hierarchically decomposed as this reflects both the structure of most organizations and the natural disaggregation of decisions, as more information becomes available as processes progress. A drawback, however, is that these frameworks are often explicitly oriented towards particular industries, like, for instance, production planning (Hax and Meal, 1973) or process planning (Marri et al., 1998). Zijm (2000) argue that this delimitation makes the frameworks inadequate in practice.

Hans et al. (2012) ends on a similar conclusion. Even though there exist several frameworks regarding planning and control within the health care sector, like Vissers et al. (2001), these frameworks are weakened by their delimitation and the neglect of integration with other managerial areas. Hans et al. (2012) proposes a framework that integrates both the managerial areas involved in health care delivery and the hierarchical levels of control often present in larger organizations. This framework is presented in Figure 3.1. Even though the MSP is not limited to health care problems, the framework of Hans et al. (2012) is general enough to embrace key planning and control elements of other industries as well, making it relevant to the MSP.

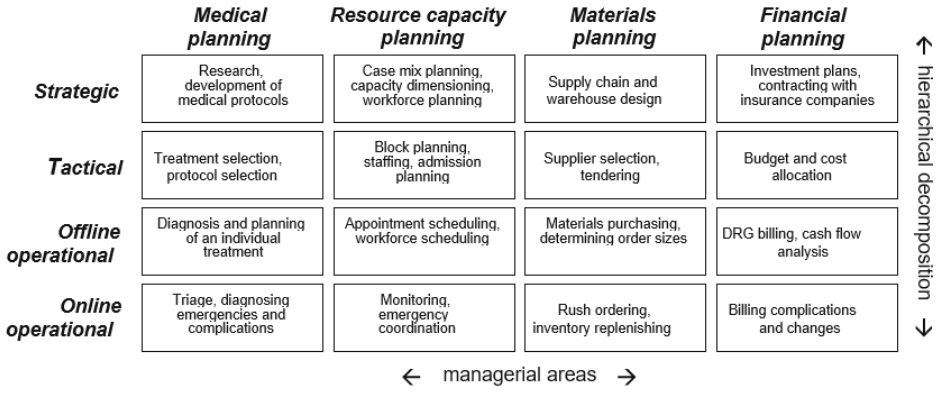

Figure 3.1: A framework for health care planning and control, (Hans et al., 2012)

The framework is divided into four managerial areas. Medical planning concerns decisions related to medical protocols, treatment and diagnosis, while the resource capacity planning addresses dimensioning, planning, scheduling, monitoring and control of renewable resources, like equipment, facilities and staff. Materials planning, on the other hand, concern the acquisition, storage distribution and retrieval of all consumable resources, such as suture materials, prostheses, blood, bandages and food. The last area, financial planning, addresses how an organization should manage its costs and revenues to achieve its objectives under current and future organizational and economic circumstances (Hans et al., 2012). The MSP is positioned within the managerial area of resource capacity planning.

The framework is also divided into four hierarchical levels. The strategic level includes structural decision making and often involves long planning horizons and a high level of uncertainty. It is the highest hierarchical level, and decisions made here influence the decisions at the lower levels. The tactical level typically has a shorter planning horizon than the strategic level, but a longer horizon than the two operational levels. It addresses the organization of the operations of delivery, like block planning and staffing. Finally, the two operational levels involve short-term decision-making related to the execution of certain processes. The difference is that the offline operational level is in advance of a planning process, while online operational level concerns decisions while a process is going on, aiming to react to unforeseen or unanticipated events. Both of the operational levels have

rather low flexibility, as the decisions are limited by decisions made at the higher hierarchical levels (Hans et al., 2012).

The MSP is best positioned within the offline operational level. The MSP regards decisions made on the strategic and tactical level as fixed input and does not take into account reactive measures to account for unforeseen events that occur during a planning period, like in the online operational level. The main purpose of the MSP is to generate a schedule that allocates a fixed workforce to cover forecasted demand that is to be treated fixed.

3.1.2 Classification of Personnel Scheduling

Baker (1976) was one of the first to classify personnel scheduling by distinguishing three separate main elements: shift scheduling, days-off scheduling and tour scheduling. While shift scheduling concerns when staff is to work during a single day and days-off scheduling address the allocation of off-days, a tour schedule is classified as the combination of the two aforementioned elements. In a tour schedule, the time and day an employee is to work or not work are determined for a more extended period of time, and this pattern of when an employee is working and not constitutes a *tour*.

In later years, a number of attempts have been made to classify personnel scheduling further. Some of the most acknowledged works are Tien and Kamiyama (1982), Ernst et al. (2004), De Causmaecker et al. (2004) and Van Den Bergh et al. (2013). These works will not be discussed in detail, but they provide a perception of the diversity of alternative classification approaches.

Bhulai et al. (2008) offers an alternative classification approach, highly relevant to this master's thesis. In this work, the personnel scheduling is classified by four phases; (1) workload prediction, (2) staffing, (3) shift scheduling and (4) rostering. Workload prediction simply concerns the prediction of future workload, while staffing aims to translate this workload into a required number of satisfactory qualified personnel such that a predefined service level is met. Shift scheduling, on the other hand, is concerned with the process of generating and creating shifts so that the desired staffing levels are obtained. Finally, the rostering phase pair created shifts to employees in order to obtain desired demand coverage (Bhulai et al., 2008).

The classification approach proposed by Bhulai et al. (2008) offers a relevant framework for the classification and positioning of the MSP. The MSP is concerned with both shift design and rostering, equivalent to phase (3) and (4) in Bhulai et al. (2008). In these phases, the main objective is to design shifts based on the available workload predictions and staffing, and later roster employees to these designed shifts. This is the same approach that we have selected in order to solve the MSP. The tasks of creating shifts and rostering the employees are further referred to as the shift design problem and the rostering problem, respectively, and the combination of these two problems is what we hereby define as personnel scheduling.

3.2 Literature Search

Based on the positioning of the MSP in Section 3.1, the literature review is further concerned with literature covering shift design problems and rostering problems within the offline operational level. In this section, we introduce an overview of the literature we consider most relevant to the MSP and which forms the basis for the remainder of this chapter.

The literature search is primarily based on the reviews on personnel scheduling literature in E. K. Burke et al. (2004), Ernst et al. (2004), and Van Den Bergh et al. (2013), providing great insights into modeling choices and solution approaches within the world of personnel scheduling. Furthermore, a specialized search was completed in order to uncover articles, explicitly addressing the various aspects of personnel scheduling problems. This concerns literature targeting the shift design problem or relevant solution approaches for the rostering problem in particular. For this purpose, the academic search engine Google Scholar was utilized, applying targeted keywords or exploring cited articles of related work.

The literature we consider most relevant to this thesis is presented in a non-prioritized order in Table 3.1. It is to be noted that additional literature, beyond what is presented in Table 3.1, will be addressed. However, this will only include literature that is considered supplementary or useful for understanding a broader context.

Table 3.1: Overview of literature considered most relevant to the MSP

1	E. K. Burke et al. (2004)	16	E. K. Burke and Curtois (2014)
2	Van Den Bergh et al. (2013)	17	Lin et al. (2015)
3	Di Gaspero et al. (2005)	18	Beckmann and Klyve (2016)
4	Musliu et al. (2004)	19	Thompson (1995)
5	Gärtner et al. (2007)	20	Lapègue et al. (2013)
6	Nilssen et al. (2011)	21	Prot et al. (2015)
7	Di Gaspero et al. (2013)	22	Geitle et al. (2019)
8	Jaumard et al. (1998)	23	E. Burke et al. (2003)
9	Felici and Gentile (2004)	24	Lü and Hao (2012)
10	Azaiez and Al Sharif (2005)	25	Stølevik et al. (2011)
11	Bard and Purnomo (2005)	26	Smet and Vanden (2016)
12	Wright et al. (2006)	27	E. K. Burke et al. (2010)
13	Rönnberg and Larsson (2010)	28	Ropke (2009)
14	Jenal et al. (2011)	29	Pillac et al. (2013)
15	Ouelhadj et al. (2012)		

3.3 Shift Design Strategies

In this section, we present the most common strategies for designing shifts. When a demand forecast is available, the process of shift design could be initiated. The overall goal of this process is to create shifts that employees can be assigned to, such that the forecasted demand is covered and legislation and regulations are satisfied. This is what we have defined as the shift design problem. According to the work of Musliu et al. (2004), there exist two main approaches to solve the shift design problem. One approach is to tightly coordinate the process of creating shifts with the process of allocating employees. In this way, the shift design problem and rostering problem, as we have defined it, are solved as a single problem. The other approach is to split these two processes, solving them in separate stages. This implies that the shifts are designed based on the demand forecast and staffing level, and then the rostering is based on those pre-designed shifts. The first approach typically provides better solutions as it closer relates the shifts to the actual rostering process, i.e., enabling better adaption to employee aspects like contractual agreements and preferences. The second approach, on the other hand, is commonly easier to solve and thus better equipped to handle large and complex problems (Musliu et al., 2004). In this master thesis, we follow the second approach.

Another closely related subject within shift design strategies is whether to model the shifts explicitly or implicitly. In 1954, Dantzig was the first to address the shift design problem as part of the personnel scheduling process (Thompson, 1995; Aykin, 1996). Dantzig modeled the shift design problem as a mathematical set-covering problem, requiring an explicit representation of all possible shifts. A drawback of using explicit shift formulations is that the number of alternative shifts increases rapidly as the flexibility and complexity of the problem increases. When allowing for a broader range of possible shift durations, or end or start times for a shift, the number of alternative shifts increases, resulting in an escalated problem complexity. Additionally, scheduling breaks or allowing skill-specific shifts to increase the number of shifts, making the problem harder to solve (Thompson, 1995).

The alternative modeling technique, less frequently tackled in literature, is to use implicitly generated shifts in the scheduling process. When using an implicit shift formulation, the coverage of demand is not accomplished through assigning employees to predefined shifts. Employees are rather assigned a start time and a duration, that aims to cover the predicted demand. The concept of implicit shift modelling was introduced in Moondra (1976), as stated in both Thompson (1995) and Rekik et al. (2010). Moondra defined sets with a various number of shifts in each planning period. Constraints were imposed on the shifts to limit their allowable duration, facilitating a high degree of shift length and starting time flexibility (Thompson, 1995). Similar modelling techniques have been used in other works, like Brunner et al. (2009) and Isken (2004). The implicit shift modeling approach becomes increasingly valuable when scheduling breaks during a shift. The placements of breaks is a discussed topic in the literature (Gärtner et al., 2007; Rekik et al., 2010; Aykin, 1996), but is not presented in detail as the scheduling of breaks is out of the scope of this master thesis. A major drawback of implicit shift design is, however, the complexity such approaches entails, and consequently, implicit models are rarely used in practical applica-

tions.

In this master thesis, we are modeling the shifts explicitly. We do, however, draw inspiration from the implicit approach and try to imitate and transfer elements from flexible shift design into our approach to solve the shift design problem. In addition, a Shift Reduction extension is proposed to better overcome some of the challenges discussed regarding the use of explicit shifts.

3.4 Personnel Scheduling Strategies

In this section, the most common personnel scheduling strategies are presented. Within the world of personnel scheduling, there exist several different methodologies for how the scheduling could be carried out. Note that scheduling here concerns the whole personnel scheduling process as we have defined it, i.e., the combination of the shift design problem and rostering problem, but with an elevated appliance for the rostering problem. E. K. Burke et al. (2004) divides these methodologies into three modes of operation. These methodologies are centralized scheduling, unit scheduling and self-scheduling. Even though these modes of operation are discussed in the light of nurse scheduling, the authors address that the terms have been used to cover different types of personnel scheduling problems.

In the case of centralized scheduling, a small team is assigned the responsibility of being in charge of scheduling for the entire organization. Doing so relieves the heads of departments, or other employees from scheduling, enabling them to focus on their other responsibilities. The use of centralized scheduling enables better use of resources, which could lead to better cost-efficiency (E. K. Burke et al., 2004). Unit scheduling, on the other hand, assigns the task of constructing personnel schedules to the head of the department or a scheduling manager. The advantages of this method are that the planning process is moved closer to those it concerns, and thus increases the ability to take personal preferences and unit specific requirements into account. The last approach is self-scheduling, which implies that the responsibility of creating schedules is transferred to the employees. Each employee creates a schedule draft, and in cases with conflicting schedules, the employees negotiate an agreement in their spare time. This process is often done manually, making it a time-consuming approach. The advantage, however, is that the opportunity to include personnel preferences more efficiently is increased, in addition to allowing the employees to affect their schedules more closely. On the other hand, self-scheduling could lead to unfair schedules as some employees are better at bartering than others. Additionally, there is a higher chance of conflict amongst nurses (Beckmann and Klyve, 2016). A more thorough explanation of self-scheduling is given in Rönnberg and Larsson (2010).

In addition to the three modes of operation presented by E. K. Burke et al. (2004), Bard and Purnomo (2005) introduces preference scheduling as a forth approach. This approach incorporates personal preferences into the personnel scheduling process and is referred to as preference scheduling (Bard and Purnomo, 2005). In Bard and Purnomo (2005), each employee submits a list of preferred shifts which is taken into consideration by the

scheduling manager. The MSP is best described by the preference scheduling approach, as the employees submit their preferences to the model as input data, which are then used to create schedules that maximize the satisfaction of preferences.

Another aspect that influences the personnel scheduling process is whether schedules are cyclic or non-cyclic (E. K. Burke et al., 2004). A cyclical schedule entails assigning the employees to work a cycle of a predefined number of weeks. This approach has been explored in some detail in literature and has been used in real-world applications (Rocha et al., 2013; Jenal et al., 2011). The idea is that a repeating and non-changing pattern of demand could be handled with a repeating and non-changing pattern of shift schedules, i.e., cyclic schedules. A pattern of working days is identical to each employee, but where these patterns are slightly shifted between the employees. Such schedules facilitate predictability and are commonly more easy to solve as the final schedule can be repeated several times (Jenal et al., 2011). In the case of non-cyclical schedules, a new schedule is created for each planning period, and there is not necessarily any repeating or identical pattern among the employees. Such schedules enable a better fit to employee preferences and often provide increased flexibility. In regards to cyclical and non-cyclical scheduling, the MSP is best defined as the latter. This is important as our general model must be able to handle both cyclic and non-cyclic characteristics.

3.5 Key Concepts of Personnel Scheduling Problem

This section elaborates on the most important concepts of the scheduling problem and how they are treated in the literature. The concepts are grouped thematically below.

3.5.1 Shift Design

As stated in Nilssen et al. (2011), a prerequisite for high-quality rosters is an efficient shift design. Shift design is concerning the process of finding a set of shifts that match a predicted demand, and that additionally satisfies a set of constraints. These constraints could, for instance, be related to legal requirements or organization-specific desires on the characteristics of the shifts. Also, a common aim of shift design is to realize some objective, like minimizing costs or increase staff and service satisfaction (Nilssen et al., 2011).

Shift Types and Flexibility

The majority of literature on personnel scheduling base its models on predefined shifts (Thompson, 1995), the same as explicit shift representation as discussed in Section 3.3. It is most common to deal with a small set of predefined shifts to cover demand (Prot et al., 2015), where the most common shift duration within the health care industry are between 8- and 12-hour (Bard and Purnomo, 2005). This has led to the standard shift distribution already presented in Figure 2.3 in Chapter 2, where shifts categorized as Day, Evening and Night are used to cover 24 hours. It is also common to extend the shift distribution by including shifts categorized as Morning, offering an intermediate step between the Night and

Day shifts (Musliu et al., 2004; Gärtner et al., 2007; Di Gaspero et al., 2005; Di Gaspero et al., 2013).

We see the use of these standardized shifts in Jaumard et al. (1998) and Bard and Purnomo (2005), but there also exists variations of these shifts in a number of other works (Lin et al., 2015; Azaiez and Al Sharif, 2005; Wright et al., 2006). The work of Jenal et al. (2011) is however arguing that the use of 7-, 9- and 10-hours shifts allow for increased flexibility, while Prot et al. (2015) takes this even further by generating and allocating workers to customized shifts as the first step in a two-stage problem. Here, the maximum shift duration is set to 11 hours but other than that a shift can be of any duration. Flexibility in shift design is argued to be vital to minimize the mismatch in employee supply and demand (Porto et al., 2019), and real-life application of scheduling flexibility has proven to have a very positive impact on labour utilization (Rekik et al., 2010). It is to be noted, however, that the incorporation of scheduling flexibility increases the complexity of the rostering process. The work of Musliu et al. (2004) facilitates flexibility by defining certain intervals for when shifts are allowed to start, and with a corresponding limit on the minimum and maximum shift duration. Their approach is visualised in Table 3.2.

Table 3.2: An example of four shift types used in Musliu et al. (2004)

Shift type	Min-start	Max-start	Min-length	Max-length
Morning	05:00	08:00	07:00	09:00
Day	09:00	11:00	07:00	09:00
Evening	13:00	15:00	07:00	09:00
Night	21:00	23:00	07:00	09:00

By allowing intervals of both possible starting time and shift duration, Musliu et al. (2004) can more closely adapt the designed shifts to the specific problems, allowing for increased flexibility.

In our approach of solving the MSP, we design shifts similar to Prot et al. (2015) and Musliu et al. (2004). We operate with a lower limit of five hours for the shift duration, while 12 hours is the maximum limit. This provides extended flexibility compared to the aforementioned works, even though Prot et al. (2015) does not have any lower bounds on shift duration. In addition, we are not considering fixed starting intervals for shifts, like done in Musliu et al. (2004). This is done to increase the flexibility further and to enhance the generality of the model.

Shift Design Objectives

The most common objective component in shift design is to minimize the deviation between the applied workforce and demand (Musliu et al., 2004; Nilssen et al., 2011). As discussed in Chapter 2, a deviation between demand and the number of employees at work lead to under- or overstaffing, affecting both the work performance and costs. There are also some literature covering other cost aspects as well, as the cost associated with each

specific shift (Thompson, 1995; Nilssen et al., 2011; Musliu et al., 2004). Distinguishing the cost between different shifts is favorable as the cost level will depend on aspects like the time of the day and the required competence level. Another common aim is to minimize the number of possible shifts. Reducing the number of shifts has several advantages, as it facilitates team-building as persons are kept together, and the complexity of scheduling, managing and control are reduced considerably (Di Gaspero et al., 2005). Additionally, there exist examples of literature including more problem-specific objectives, like minimizing the average number of duties per week (Musliu et al., 2004; Di Gaspero et al., 2013) or to minimize the the gap between the targeted clinical workload and assigned clinical workload for each employee (Lapègue et al., 2013).

In solving the MSP our main objective is to create shifts that enable minimization of deviation between applied workforce and demand. In addition, we propose a Shift Reduction extension that has the objective of reducing the number of alternative shifts and minimize the use of shifts that are considered of a non-ideal duration by the user. The latter implies that the user can allow non-ideal shift durations to facilitate ideal demand coverage, and this is, as far as we know, not covered in the existing literature.

3.5.2 Demand Coverage Modelling

Within the demand dimension of the scheduling problem, there exist several important aspects and considerations. One of them, the process of designing shifts, are thoroughly discussed in the previous subsection. Besides shift design, two distinct aspects stand out, which are: coverage of demand, and the relation between shifts, demand and time. The critical aspect of demand coverage is to allocate a sufficient amount of employees to shifts in such a way that they can meet the demand for each day throughout the planning period. The second aspect, on the other hand, entails the important consideration of how shifts, demand, and time are related.

Coverage of Demand

It is most common to model the coverage of demand as a hard constraint (Van Den Bergh et al., 2013). Lin et al. (2015) and Felici and Gentile (2004) have hard constraints stating that the minimum demand must be met. Bard and Purnomo (2005) and E. K. Burke and Curtois (2014) include both a minimum and maximum demand, modeled as hard constraints. Wright et al. (2006) has a unique way of modeling demand. They simulate demand for each shift and allocate employees to work based on a patient-to-nurse ratio, in combination with a hard restriction on the minimum expected service level. This makes it possible to violate the demand restriction for short periods. The MSP requires a hard constraint on both minimum and maximum demand but also aims to minimize deviation from a defined level of ideal demand.

A hard constraint on satisfying demand can make the problem highly constrained or even infeasible. As a consequence, several authors include measures to relax the problem. Bard and Purnomo (2005) allows for the use of temporary employees as a way of relaxing the minimum demand constraint, while Azaiez and Al Sharif (2005) allows for overtime. Prot

et al. (2015) allows for overtime to meet the demand, and as a last resort, external nurses can be used. Wright et al. (2006) allow for scheduled overtime in increments of whole shifts to increase flexibility. The MSP allows for weekly overtime, but the accumulated working hours for each employee has to be equal or less than the accumulated contracted hours over the entire planning period.

As discussed in Chapter 2, the demand could also be linked to specific competencies. There exist two common ways to handle the demand for specific competencies within the nurse scheduling literature: categorical skills and hierarchical skills (Van Den Bergh et al., 2013). The categorical skill approach is based on defining a set of different skills. As long as a nurse possess any particular skill, the nurse may cover any demand for that particular skill. There is no difference in skill levels among the nurses, beyond having a particular skill or set of skills. This approach is seen in works like Jenal et al. (2011), Ouelhadj et al. (2012) and Prot et al. (2015). The hierarchical approach, on the other hand, regards a nurse with a higher skill level as more educated and experienced than a lower level nurse. Thus a higher level nurse can do all the tasks a lower level nurse can perform, but not the other way around. Jaumard et al. (1998), Rönnberg and Larsson (2010) and Beckmann and Klyve (2016) all uses hierarchical competencies. The categorical skill approach is used for the MSP.

Relation Between Shift, Demand and Time

In most of the shift design literature discussed in Subsection 3.5.1, the demand is expressed directly through the shift (Di Gaspero et al., 2013; Gärtner et al., 2007; Nilssen et al., 2011; Musliu et al., 2004). The concern is thus to allocate enough staff to the various shifts.

An alternative approach is to define the need for employees as a function of time, rather than as a function of shifts. This is often the basis for flexible shift design, discussed in Section 3.3, and is the approach used in Thompson (1995), Prot et al. (2015) and Lapègue et al. (2013). Prot et al. (2015) models demand as a set of tasks with a given start time and duration that has to be carried out by one worker. The timing of the tasks does not necessarily conform to any predefined shift, and thus the need for flexible shifts arises for their problem. In Lapègue et al. (2013), there are no fixed shifts defined. Employees could start and end their working days whenever it is necessary, provided the resulting shift adhere to the set requirements.

In the MSP, the demand is time-related. The demand is made up of periods of equal length, stating the required competencies and required number of employees. When the shifts are created, they are designed in order to cover specific sequences of these periods. We are, however, not stating the number of employees required to work the required shifts. In this way, a direct relationship between time and demand is still retained, and shifts are used in order to connect employees to the time-specific demand. This is done in order to facilitate increased flexibility, discussed more thoroughly in Chapter 5, and is a novel approach not addressed in the existing literature.

3.5.3 Time-related Constraints

Time-related constraints are a collection of similar constraints regarding working hour regulations defined by the government and labor union agreements, as well as preferred organizational practices. Typical for these constraints are their aim to prevent overworked and fatigued employees, and promote as good working conditions as possible.

Working Hours Regulations

One of the most common constraints in literature is to enforce at most one shift per day (Prot et al., 2015; Wright et al., 2006; Rönnberg and Larsson, 2010; Azaiez and Al Sharif, 2005; Beckmann and Klyve, 2016). As seen in Subsection 3.5.1, it is also common to only allow shifts of specified duration, generally between eight and 12 hours. Besides, there might be restrictions on the weekly working hours, like 48 hours used in Prot et al. (2015) or 36-40 hours, depending on the contract of the employee in Wright et al. (2006). In the MSP, we only allow one shift per day. We do, however, allow the user to define what day a shift belongs to, implying that an employee could be able to start two shifts during a calendar day. This is done in order to allow night shifts, that for instance span form 22:00 to 08:00, to be considered shifts belonging to the day where the majority of the working hours are worked. When it comes to the allowed shift durations, this is provided as an input from the users. The MSP does not impose an explicit limit on the weekly worked hours, but overwork is indirectly handled by requiring sufficient daily and weekly rest, as well as only allowing one shift per day. Moreover, each employee is restricted to not work more than the accumulated contracted working hours for the entire planning period.

Rest and Time-Off

To achieve even distribution of the work within a planning period, the most common approach is to regulate how many consecutive days an employee can work. Azaiez and Al Sharif (2005) uses a four day limit on working consecutive days, while Rönnberg and Larsson (2010) operates with a limit of five days. Jenal et al. (2011) and Prot et al. (2015) have a limit of six consecutive working days. Ouelhadj et al. (2012), Bard and Purnomo (2005) also implements such a constraint. Azaiez and Al Sharif (2005) include that a nurse should get four weekend-days off during a 4-week period, while also having 14-16 working days during the same period. Jenal et al. (2011) allows between 12 and 14 working days per schedule, which consists of 21 days. Felici and Gentile (2004) have a hard constraint demanding that each nurse is allocated to exactly two rest shifts per week.

A minimum rest duration of 11 hours between shifts is enforced in Prot et al. (2015), Ouelhadj et al. (2012) and Beckmann and Klyve (2016). Lin et al. (2015) uses a minimum of 16 hour rest between two shifts. Prot et al. (2015) utilizes another approach where a minimum weekly rest of 35 hours is required. In the MSP, both a certain amount of daily and weekly rest is required. The weekly rest is handled by the allocation of a weekly resting shift, while the daily rest is ensured implicitly by forbidding combinations of shifts that would violate this requirement. The required length of the daily and weekly rest is a user-defined input.

Weekends

Working weekends is something many employees prefer to avoid, but if they do have to work weekends, they prefer working a whole weekend, i.e., both Saturday and Sunday. Azaiez and Al Sharif (2005), Rönnberg and Larsson (2010), Beckmann and Klyve (2016) and Bard and Purnomo (2005) all incorporate either a soft or a hard constraint regarding partial weekends. Rönnberg and Larsson (2010) expand on the concept of partial weekends, and requires that a nurse should work Friday or Monday if he works a weekend. All of the papers strive to distribute weekend work evenly among the employees. In our approach to solving the MSP, we are modeling weekend work as a soft constraint, penalizing allocation of partial weekends.

Contracted Hours

Azaiez and Al Sharif (2005) requires that each nurse have to work at least all their contracted hours over the planning horizon. Rönnberg and Larsson (2010) sets a hard limit on max 10% deviation from contracted hours over the entire planning period, in combination with between 60% less or 50% more than the weekly contracted hours for each week in the planning period. Bard and Purnomo (2005) includes a hard constraint for minimum worked hours, while Wright et al. (2006) includes a lower and upper bound on the number of shifts an employee can work each week, and thus ensuring that contracted hours are reached. In the MSP, the employees are restricted to not work more than their accumulated contracted hours for the entire planning horizon. We are, however, penalizing deviation between worked and contracted hours, and the schedules should, therefore, aim to allocate work so that the employees work close to their contracted hours.

Other Preferences

Isolated working days, meaning the pattern of having a workday between two off-days, is considered an undesired pattern. The same goes for isolated off-days, which is the opposite of isolated working days. Both these patterns are included as soft restrictions in Azaiez and Al Sharif (2005) and Bard and Purnomo (2005). Jenal et al. (2011) only included isolated working days as a soft constraint. In our solution approach to the MSP, we account for both isolated working days and isolated off-day, modeling them as soft constraints.

3.5.4 Fairness

Fairness is becoming an increasingly important aspect of personnel scheduling. Bard and Purnomo (2005) emphasizes the importance of fairness in health care by stating that bickering, increase absenteeism, low morale, poor job performance, and high turnover rates may become the norm if nurses feel like their preferences are not taken into account over longer periods of time. Smet et al. (2012) states that the quality of working schedules affects the job satisfaction of the employees and that fairness has previously been neglected in many papers. Interestingly, there exists a wide range of definitions and interpretations of fairness. Bard and Purnomo (2005) points to fairness as equaling a balanced distribution of the desirable and undesired aspects of schedules. Rönnberg and Larsson (2010), on

the other hand, operates with a core component of fairness. This includes maximization of personal preferences, in combination with ensuring and that no single employee has to put up with a low-quality schedule relative to the others. Meanwhile, many authors do not explicitly address fairness as a term, but as Beckmann and Klyve (2016) states that "most models include constraints that are obviously concerned with it [fairness]".

Personal Preferences

Personal preferences are individual desires regarding the schedule. This is usually limited to a preference for working or not working at specific times or shifts (Beckmann and Klyve, 2016; Bard and Purnomo, 2005; Wright et al., 2006). Rönnberg and Larsson (2010) allows the employees to state soft and hard requests regarding preferred times for work and off. For the hard requests, each employee is given a set of vetos that has to be met. The soft requests, on the other hand, are preferences that a schedule ideally fulfills, but which does not necessarily have to be fulfilled. In Rönnberg and Larsson (2010), the employee can weight its soft request by reducing the number of preferences, so that the likelihood of fulfillment is increased. We use a similar approach as in Rönnberg and Larsson (2010), but where the concepts of vetos are handled implicitly in the contractual agreements as blocked hours.

Weighting of Fairness

In the opinion of Smet et al. (2012), a weighted sum in the objective function is suboptimal as it "does not explicitly account for the fair distribution of individual high quality rosters". They further state that models based on this approach can lead to unfair solutions as the majority of the employees may get a high-quality solution at the expense of a few employees, which is allocated to low-quality schedules. Grano et al. (2009) uses an auction-based system to accommodate fairness for nurses. This practice has been criticized for being unfavorable as some nurses may be better at negotiating (Smet et al., 2012; Bard and Purnomo, 2005). E. K. Burke and Curtois (2014) uses a numbering system to penalize violation of preferences and desires. Bard and Purnomo (2005) cites experience when noting that solution that relies too much on the user to quantify preferences have trouble with being accepted by the businesses. Rönnberg and Larsson (2010) also has trouble with the employees not using all the available tools to express their preferences. To achieve the wanted balance in the schedules, Bard and Purnomo (2005) proposes that the number of violation of preferences should be approximately equal for every nurse. Their works also include means to ensure a balance scheduled, using a system where each nurse allocates penalty points to each possible violation of a desired schedule. Lastly Bard and Purnomo (2005) expand on E. K. Burke and Curtois (2014) by creating a more complex fairness model, where they defined the cost coefficient c_{ej} as an increasing function based on the number and severity of the violated constraints, as illustrated in Figure 3.3.

In the MSP, the user is enabled to set the weights based on desires and priorities. As the implemented solution methods aim to both minimize demand deviation and maximize fairness, we propose an approach where the weights of the various fairness aspects are

Table 3.3: Bard and Puronmo's cost coefficient for fairness violations

Degree of violation	No. of penalty points, v	Cost coefficient, $c_{ij}(v) = 2^{v-1}$
Simple	1	1
Serious	2	2
Severe	3	4
Extreme	4	8

weighted relative to the weight of demand deviation. This allows the user to state the relative weight between demand deviation and fairness. Also, the approach facilitates relative weighting between the various fairness aspects.

Fairness Objectives

There exists a variety of different fairness objectives for scheduling problems, and most of the authors reviewed in this section have some form of fairness included in their objective. Lin et al. (2015) maximizes preferences for both desirable and undesirable working times, while Bard and Purnomo (2005) sets a lower limit on the number of personal preferences granted as a hard constraint. Beckmann and Klyve (2016) tries to grant as many shift request from the employees as possible.

Lin et al. (2015) only focuses on preferences by maximizes the number of preferences granted, while Prot et al. (2015) and Felici and Gentile (2004) maximizes staff satisfaction for allocated shift patterns. Wright et al. (2006) uses a bi-criteria formulation where they minimize labor costs and maximize schedule desirability. Rönnberg and Larsson (2010) base their fairness component on personnel requests. They maximize the number of granted requests, in combination with an additional reward for the least favored nurse and thereby facilitating increased quality of the worst schedule. In this thesis, we use a similar approach as in Rönnberg and Larsson (2010), where the evaluation of the fairness score of the entire staff as well as for the least favored employee are key contributors to the objective function. By including the score of the least favored employee, a more even fairness distribution is facilitated. Other fairness aspects, like preferences as well as certain working patterns, partial weekend and long weekly rest periods are all included as they constitute the fairness score for each employee.

3.6 Optimization Approaches

To solve the personnel scheduling problem, a vast number of different algorithms and solution methods are proposed throughout the literature. Van Den Bergh et al. (2013) presents a categorization of 7 different main categories. Within these categories, the majority of papers are classified as either mathematical programming, construction heuristics, or improvement heuristic. These three categories define a classification of their own: exact (mathematical) and meta (heuristics). In most of the literature referenced in this section,

the terms metaheuristics and heuristics are used interchangeably. The terms will therefore be used interchangeably in this section.

3.6.1 Mathematical Programming

A personnel scheduling problem solved using mathematical programming are usually modelled linearly, as either a integer or mixed integer problem. The advantage of mathematical programming lies in its ability to reduce the search space, generate exact optimal solutions and prove optimality. However, the complex nature of real-world personnel scheduling problems poses a limitation that makes it difficult to formulate a mathematical model. Additionally, most real-world scheduling problems are considered NP-complete. This complexity often result in simplifications of the problem, making the resulting model less relevant in practical applications. A number of different approaches have been proposed to solve rostering problems. As set-covering models of scheduling problems tend to result in a high number of variables, decomposition techniques are commonly applied to overcome these large scale formulations Van Den Bergh et al. (2013). One such approach is column generation, which strength lies in not needing to hold all variables in the model at once. Brunner and Edenharter (2011) and Bard and Purnomo (2005) both apply column generation to problems resembling the MSP. Branch-and-price is another approach, utilizing column generation in combination with branch-and-bound, specifically designed to solve large scale integer programming problems (Savelsbergh, 2001). This approach have been investigated in a number of articles (Brunner et al., 2011; Ni and Abeledo, 2007; Purnomo and Bard, 2007; Stark and Zimmermann, 2005). As the main contribution of this thesis is the implemented heuristic, mathematical approaches are not further elaborated upon.

3.6.2 Metaheuristics

For a variety of combinatorial optimization problems, metaheuristics are regarded the most effective optimization approach (Arabia and Alfares, 2004). The benefits of metaheuristics lie in their ability to often generate feasible solutions faster, compared to exact solution methods. Typical application areas are problems with real-world complexity, where exact solutions are too computationally heavy to generate solutions within a reasonable time frame. One drawback of metaheuristics is that it cannot prove optimality of a solution, or even demonstrably reduce the search space of the problem (Van Den Bergh et al., 2013). Additionally, adding new constraints to the problem could require a reformulation of the heuristic algorithm. A vast number of different heuristic solution methods have been suggested to solve the employee rostering problem. As the field of metaheuristics is immense, we will only highlight a few interesting approaches, considered most relevant to the MSP.

Genetic algorithms are a method that has emerged in recent years as a useful tool to solve complex discrete optimization problems (Aickelin and Dowsland, 2004), and is often preferred by researchers Van Den Bergh et al. (2013). Dean (2008) implements a genetic algorithm with a two-dimensional chromosome structure to solve a nurse rostering problem consisting of 14 resident nurses, complying with hard and soft constraints. Aickelin and Dowsland (2000) and Aickelin and Dowsland (2004) also investigates the utilization

of genetic algorithms in nurse rostering problems. However, due to the highly constrained nature of the rostering problems, the algorithms proposed struggled to find feasible solutions for some of the instances. To increase the performance of the genetic algorithms on highly constrained problems, E. K. Burke et al. (2010) implemented a hybrid evolutionary algorithm that combines a genetic algorithm with a simulated annealing hyper heuristic. The article shows consistently high performance on all of their 52 test instances when compared to previously used approaches.

Another class of metaheuristics used on rostering problems is swarm optimization with approaches like Particle swarm optimization (Wu et al., 2015) and firefly optimization. Karmakar et al. (2016) compared these approached with an implementation of a simulated annealing algorithm as well as a genetic algorithm. The four solution approaches are tested on 25 instances, running for 20 seconds on each instance. The results show that firefly optimization has the best potential on the tested instances.

The last class of metaheuristics discussed is the neighborhood search heuristics, which is broadly investigated in the literature. As this thesis implements a Parallel Adaptive Large Neighborhood Search heuristic, the next section is dedicated to a more thorough review of neighborhood search, with a focus on Large Neighborhood Search, Adaptive Large Neighborhood Search, and Parallell Adaptive Large Neighborhood Search.

Theory on Neighborhood Search

The Large Neighborhood Search (LNS) was first proposed by Shaw (1997) as an approach to solving vehicle routing problems. The LNS is part of the class of algorithms called Very Large Neighborhood Search (VLNS) (Ahuja et al., 2002). The idea behind all VLNS algorithms, is that searching a large neighborhood results in finding local optima of high quality, which may lead to better solutions. However, the search is often time consuming. To mitigate this effect, the algorithms take advantage of various filtering techniques, limiting the search to a subset of the neighborhoods. Large Neighborhood Search is based on relaxing and re-optimization of an initial solution. This is done through operators that destroy part of the current solution, then tries to rebuild it into a more optimal solution. These operators are generally very powerful and have the potential of altering a large part of the solution. The concept of improving a solution through destroy and repair operators have occurred in computer science under other names like fix and optimize (Pisinger and Ropke, 2007) and ruin and recreate (Schrimpf et al., 2000), which is the inspiration for the LNS. As the algorithm always chooses the best solution, it is denoted a best improvement algorithm. This is opposed to a first improvement approach that chooses an updated solution from the first direction of descent.

Adaptive Large Neighborhood Search (ALNS) extends the LNS by allowing for multiple destroy and repair operators. The algorithm was first introduced in Ropke and Pisinger (2006) as an improved method of solving the pickup and delivery problem with time windows (PDPTW). The idea is that the success of destroy and repair heuristics might vary

depending on the problem instance. Additionally, finding a suitable heuristic for each problem instance might prove difficult. The ALNS allows multiple destroy and repair heuristics to be implemented, coupled with a weight indicating their previous success. To select the operators, the roulette wheel selection principle is used. To ensure the most advantageous operators are used more often, the weights are adjusted dynamically based on their performance.

Application of Neighborhood search Heuristics in Personnel Scheduling

Lü and Hao (2012) implements an Adaptive Neighborhood Search (ANS) that adaptively switches between three search strategies according to a diversification level. Two different moves are proposed; moving one shift at a specific day to a different nurse, and swapping the shift of two nurses working the same day. These moves are denoted one-move and two-swap, respectively. Additionally, a tabu search is implemented in the intensive search strategy to help explore the entire neighborhood produced by the moves. The process of dynamically switching among different search strategies in accordance to a diversification level, share some similarities to the PALNS proposed in this thesis. However, Lü and Hao (2012) does not change neighborhood between the search strategies, which is a crucial component to the PALNS. Additionally, the algorithm explicitly defines the neighborhoods through its moves, in contrast to the PALNS, where the neighborhoods are defined implicitly, through destroy and repair operators.

A Variable Neighborhood Search (VNS) was implemented in E. K. Burke et al. (2003), modeled after Belgian nurse scheduling problems, which are highly constrained. The authors present a number of different neighborhood structures based on specific problem information, and the search is executed through tabu search and steepest descent. They observed it to be beneficial to combine various neighborhoods with a simple search heuristic, as it were able to better explore the search space, as opposed to more sophisticated heuristics. Furthermore, a number of articles research the effects of hybrid approaches, based on the VNS. Stølevik et al. (2011) develop a hybrid solution approach between Constraint Programming, Iterated Local Search, and Variable Neighborhood Descent (VND). To escape local optima and ensure diversification, the authors employ ruin and recreate methodology. This is done through the removal of between 2% and 30% of the shift assignments. The new partial solution is treated as input to their CP search, which constructs a new feasible solution. The approach was developed to address the nurse rostering needs of health care institutions in Norway and Sweden. The problem shares similarities with the MSP studied in this thesis. However, the number of shifts considered is far less, leading to a significantly reduced solution space for the same problem size. Another hybrid approach were proposed by E. K. Burke et al. (2010). Considering the advantages and disadvantages of exact and metaheuristic solutions, the authors present a decomposition technique that combines the use of integer programming and Variable Beighboring Search to solve a highly constrained nurse scheduling problem.

Smet and Vanden (2016) presents an implementation of the LNS that aims to address

large instances of the multi-task shift assignment problem (MTSAP). This problem shows resemblance to the MSP presented in this thesis. The MSP, however, assigns demand on a time period basis instead of tasks, leading to potentially much larger solution space. The algorithm utilizes two destroy heuristics, horizontal and hamming distance, which are differentiated by which part of the solution they destroy. To repair the solution, the authors implement an integer programming model that is sped up through branching priorities.

The performance of LNS and its extension ALNS, in regards to personnel scheduling, is fairly unexplored. Throughout the search for relevant literature, the only implementation found is done by Geitle et al. (2019). They investigate the performance of a VNS and an ALNS compared to a MIP model on a nurse scheduling problem faced by one of the largest hospitals in Norway. The aim is to investigate the solution approach that achieves the best performance on a problem instance representing the hospital, within a restricted time limit. The PALNS implemented in this thesis does, however, go further by taking advantage of parallelization in the ALNS iterations. The PALNS implemented utilizes a high number of repair and destroy operators, both heuristic and optimal. Local search operators are also added to improve on potentially great solutions.

Parallelization Strategies for ALNS

It is common for a modern computer processing units (CPU) to contain multiple cores. In general, sequential algorithms runs one a single core at a time, leaving the remainder of the cores idle. Parallelization is a technique to run an algorithm in multiple parallels simultaneously, and thus increasing the performance of the algorithm with regards to the amount of computational work performed. The number of cores in a personal computer is typically four or eight, while servers can contain significantly higher numbers of cores. This indicates that there is a large potential for performance increases in parallel algorithms relative to sequential.

Among the literature on ALNS, only a few of the implementations utilizes parallelization. We denote such implementations as PALNS. Ropke (2009) has implemented a PALNS for the Travel Salesman Problem and the Capacitated Vehicle Routing Problem. The results show that an increase in the number of parallel processes leads to an approximately linear increase in speedup, that is, the optimal solution is found approximately 8 times faster when using eight subprocesses compared to using only one. Pillac et al. (2013) have a different take on PALNS, which they have implemented for the Technician Routing and Scheduling Problem. Pillac reports a 3.4 times speed up for 4 subprocesses compared to a sequential implementation. These results shows that parallelization can significantly improve neighborhood searches.

One important consideration is how to implement collaboration between the subprocesses, that is how to share data across the subprocesses in an productive manner. Ropke (2009) utilizes a shared state consisting of best and current solution, as well as the operator weights to enable collaboration between the subprocesses. This means that every subprocess works from the same basis, and that they are adapted to prefer the same operators during the search. Pillac et al. (2013) initializes each subprocesses with a candidate solu-

tion taken from a pool of promising solutions. The subprocesses work in isolation for a given number of processes iterations before they add their best solution to the pool. This is repeated for a specified number of master iterations. We introduce a third approach by operating with a shared queue of candidate solutions. At given intervals each subprocess will share their current solution by pushing it onto a first-in first-out queue, followed by popping another solution which is used as a candidate solution.

Another important consideration is how to implement concurrency control, that is, how to avoid incorrect behaviour caused by the parallelization. Figure 3.2 illustrate examples of correct and incorrect behavior.

Process A	Process B	Current solution
Read current solution		1
	Read current solution	1
New candidate solution, value 3		1
	New candidate solution, value 2	1
Candidate solution $>$ current solution		1
Update current solution		3
	Candidate solution $\not>$ current solution	3
	Current solution not updated	3

(a) Correct behaviour

Process A	Process B	Current solution
Read current solution		1
	Read current solution	1
New candidate solution, value 3		1
	New candidate solution, value 2	1
Candidate solution $>$ current solution		1
	Candidate solution $>$ current solution	1
Update current solution		3
	Update current solution	2

(b) Incorrect behaviour

Figure 3.2: Illustration of incorrect concurrency

To enforce concurrency control Ropke (2009) uses locks, which makes sure only one process can access the shared state from the moment the memory is locked until it is unlocked. Pillac et al. (2013) circumvents this problem by not sharing any state between the subprocesses, but every subprocesses must finish before a new master iteration can be performed. Our use of queue renders the use of locks unnecessary as subprocesses can freely pop the first element on the queue or push a new solution onto it, even if other subprocesses interacts with the queue.

One last consideration is how to make the parallelization as effective as possible. This is especially important when scaling to more than a handful of subprocesses. Amdahls law provides a rule-of-thumb to calculate the theoretical speedup for parallelization Gustafson (1988). It states that the speedup can be calculated by Equation (3.1), where p is the num-

ber of parallel processes and f is the serial fraction of a task. Figure 3.3 shows the the law applied to serial fraction from 1% to 5%. These plots show how even a small serial fraction can lead to greatly diminishing returns. Note that this is a theoretical consideration on the amount of work that can be performed with additional subprocesses. The speedups from (Ropke, 2009) measures how much faster the best solution is obtained, which does not necessarily possess a perfectly positive correlation with the amount of work performed.

$$speedup = \frac{1}{f + \frac{1-f}{p}} \tag{3.1}$$

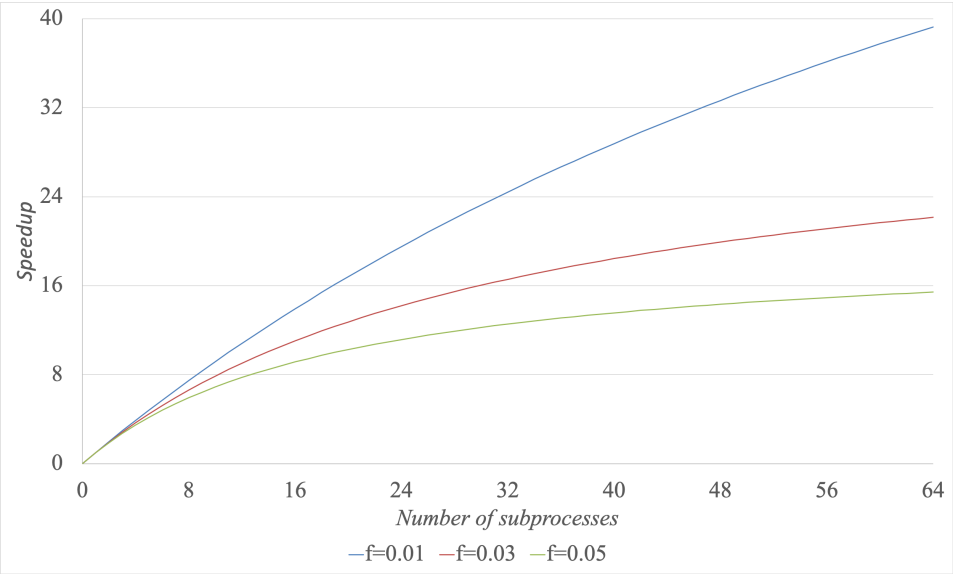

Figure 3.3: Theoretical speedups for serial fractions of 1%, 3% and 5%

Pillac et al. (2013) consists of a parallel fraction, the process iterations, and a serial fraction, the master process, as the latter process has to wait on every subprocesses to terminate before it can initiate a new set of processes iterations. Amdahls law indicates that this would not scale well if the master processes constitutes a significant part of the total runtime. The use of locks in Ropke and Pisinger (2006) results in waiting between the subprocesses. While one subprocess writes a new solution to the shared stare, other subprocesses have to wait if they want access to read it. Although subprocesses waiting on each other is not quite the same to serial code, it can be quite similar for larger number of subprocesses and short iterations. Our use of a shared queue should reduce the time subprocesses have to wait, as writing to the queue wont block anyone from reading from it.

3.7 Summary of the Literature Review

To conclude the literature review, a summary of the reviewed articles are presented systematically. The summary is based on the most relevant articles presented in Section 3.2, thoroughly discussed in Section 3.5 and 3.6. Table 3.5 compare how this master's thesis relates to existing literature regarding the key aspects of the shift design problem, while Table 3.6 compares this thesis to existing literature covering the rostering problem. Finally, Table 3.7 compare the selected solution approach to what is implemented in related works. The purpose of this summary is to clarify how this master's thesis differs from existing work. Table 3.4 presents all abbreviations used in the aforementioned tables.

Table 3.4: Explanation of abbreviations used in Table 3.5, 3.6 and 3.7

Abbreviation	Explanation
A	Area addressed. The article is either addressing (S) a specific area or industry, or (G) addressing a general area, not limited to a specific industry
$\underline{B}$, $\overline{B}$	Requirements regarding minimum and maximum contracted hours worked
B	Deviation from contracted hours
C	Cost
CD	Consecutive days
CF	Collective fairness
COM.	Competencies, can be treated as (Hie.) hierarchical or (Cat.) categorical
$\underline{D}$, $\overline{D}$	Minimum and maximum demand
DD	Demand deviation
DR	Daily rest
F	Fairness
FS, FE	Fixed start time and fixed end time for shifts
IF	Individual fairness
IO, IW	Isolated off-day and isolated work day
NES	Shift stating the number of employees required to work that shift
NS	Number of shifts
OT	Overtime allowed
P	Preferences
PA	Patterns
PR	Problem. The article is either addressing (SD) the shift design problem, (R) the rostering problem or (B) the combination of both these problems
PW	Partial weekend
RB	Required amount of rest between shifts
WR	Weekly rest
y / n	Yes / No
-	Illogical to answer how the article relates to the considered aspect
H / S	Aspect treated by hard / soft constraint
H&S	Aspect treated by both hard and soft constraints
Par	Used if an aspect is partially addressed. Used if it is misleading to say that the aspect is not included, but also misleading to say it is

A key takeaway from Table 3.5 is how similar our implemented solution approach is to

literature applying implicit shift design models when it comes to shift types and flexibility, even though the solution approach is not implemented using implicit shifts. This is in line with what is briefly discussed in Section 3.3, where it is stated that we in this thesis draw inspiration from the implicit design approach. Another interesting notion is the way we consider the objective of minimizing the number of generated shifts. Unlike other literature, we have chosen an approach where the user is able to choose whether or not the number of shifts generated should be an objective aspect.

Table 3.5: Comparison of reviewed key aspects of the shift design problem

Article	Focus		Implicit	Shift types and flexibility					Shift Objective			
Nr.	A	PR	shifts	Duration	FS	FE	NES	COM.	C	DD	NS	Other
3	G	SD	n	[7 - 9] hours	y	y	y	n	n	y	y	n
4	G	SD	n	[7 - 9] hours	y	y	y	n	y	y	y	Average duties
5	G	SD	n	[8 - 9] hours	y	y	y	n	n	y	y	Average duties
6	S	SD	n	[6 - 10] hours	y	y	y	y	y	n	y	n
7	G	SD	n	[8 - 9] hours	y	y	y	n	n	y	y	Average duties
19	G	B	y	[5 - 7] hours	n	n	n	n	y	Par.	n	n
20	S	B	y	<11 hours	n	n	n	y	n	n	n	Clinical workload
21	G	B	y	<11 hours	n	n	n	y	n	n	n	Equity
MSP	G	B	n	[5 - 12] hours	n	n	n	n	n	y	Par.	n

In this table, COM. represents whether competencies or skills are considered when creating shifts

In Table 3.6, it is clear that this master's thesis differs from existing literature within several areas. The most obvious differences are seen within the areas of rest and handling of fairness aspects. Within handling of rest, we are, to our knowledge, the first to address daily rest individually tailored to each employee. In addition, we are the first to reward long weekly off shifts, thus aiming to provide as long weekly rest shifts as possible. On the other hand, in regard to the fairness aspects, the table shows that we address a wider range than present in the existing literature. Although some of the articles include a few of the fairness aspects through the declaration of unwanted shift patterns, there is little doubt that this master's thesis has a rather comprehensive and unique attention towards the concepts of fairness.

Table 3.6: Comparison of reviewed key aspects of the rostering problem

Article	Focus		Demand			Contracted			Rest			Fairness Aspects							Fairness objective		Objective		
Nr.	A	PR	$\underline{D}$	$\overline{D}$	COM.	$\underline{B}$	$\overline{B}$	OT	RB	DR	WR	CD	B	PW	IW	IO	PA	P	CF	IF	C	DD	F
8	S	R	H	n	Hie.	-	S	y	-	-	-	n	n	n	n	n	n	S	y	n	y	n	y
9	G	R	H	n	n	n	n	-	H	n	Par.	H	n	n	n	n	H	n	y	n	n	n	y
10	S	R	H	n	n	H	n	y	H	n	n	H	n	n	S	S	S	n	y	n	n	n	y
11	S	R	H	H	n	n	n	y	Par.	n	n	S	n	n	S	S	H	S	y	n	y	n	Par.
12	S	R	H	n	n	n	H	y	H	n	Par.	n	n	n	n	n	H	n	y	n	y	n	y
13	S	R	H	n	Hie.	H	H	y	n	n	n	H	Par.	n	n	n	H	H&S	y	y	y	n	y
14	S	R	H	n	n	Par.	Par.	-	n	n	S	H	S	n	n	H	S	n	y	n	n	n	y
15	S	R	S	S	Cat.	n	S	-	S	n	n	n	n	n	n	n	S	n	n	y	n	y	y
16	S	R	H	H	n	H	H	n	Par.	n	n	S	n	n	n	n	H	n	y	n	n	n	y
17	S	R	H	n	n	n	n	-	H	n	H	n	n	n	n	n	H	y	y	n	n	n	y
18	S	R	H	H	Hie.	H	H	-	H	n	n	H	n	H	n	n	H	n	y	n	n	n	y
19	G	B	H	n	n	-	-	-	-	-	-	-	-	-	-	-	-	n	n	n	y	n	n
20	S	B	H	n	Cat.	n	H	-	H	n	H	H	n	n	n	n	n	n	n	Par.	n	n	Par.
21	G	B	H	n	Cat.	n	H	y	H	n	H	H	n	n	n	n	n	n	n	Par.	n	n	Par.
MSP	G	B	H	H	Cat.	n	H	Par.	n	H	H&S	S	S	S	S	S	n	H&S	y	y	n	y	y

Table 3.7 summarizes and compares the reviewed solution approaches to the PALNS implemented in this thesis. Of the review articles, Geitle et al. (2019) is the only one that implements an ALNS approach to the rostering problem, showing some resemblance to the solution approach implemented in this thesis. However, it is up for discussion if their implementation is correctly classified as an ALNS after the description given by Pisinger and Ropke (2007). The most common acceptance criteria are best improving, which is chosen for this thesis. Additionally, this thesis is the only approach that utilizes parallel processing to increase search efficiency in rostering.

Table 3.7: Comparison of reviewed solution approaches within neighborhood search

Article	Focus		Heuristic	Acceptance Criteria	Allow Infeasibility	Tabu	Construction	Parallel Processing
	A	PR						
22	S	R	ALNS / VNS	First Improving	Partially	Yes	Greedy	No
23	S	R	VNS	Best Improving	No	Yes	Randomly	No
24	S	R	Adaptive Neighborhood Search	Best Improving	No	Yes	Randomly	No
25	S	R	Hybrid VND and CP	First Improving	No	No	CSP Solver	No
26	G	R	LNS	-	No	No	Greedy	No
27	S	R	Hybrid VNS and IP	Best Improving	No	No	CSP Solver	No
MSP	G	R	PALNS	Best Improving	Yes	No	CSP Solver	Yes

Chapter 4

Scope and Problem Description

This chapter outlines the description of the Medvind Scheduling Problem, denoted the MSP. In Section 4.1, the scope of the problem is presented with an elaboration on important considerations. Then the problem description of the MSP is presented in its entirety in Section 4.2, with an elaboration on the objective in Section 4.3. The chapter is concluded with Section 4.4, stating the problem assumption made when addressing the MSP.

4.1 Problem Scope

The purpose of the master's thesis is to solve the MSP by generating high-quality schedules in accordance with the requirements defined by Medvind. The quality of the schedules is evaluated based on the achieved fairness of the schedule and its ability to match demand closely. Additionally, the solution method for generating schedules should be as general as needed, to be applicable to the variety of application areas and industries described in Chapter 2. The solution method is to be tested on several real-life problems provided by Medvind, representing a range of different organizations and industries. In this section, the problem scope and the most important considerations will be presented.

4.1.1 Scope and Important Considerations

As an absolute minimum, the solution method for generating schedules has to generate feasible allocations of shifts based on the defined requirements of Medvind. These requirements cover all relevant governmental work regulations, agreements with trade unions and preferred practices needed to provide a feasible real-life solution. The generated schedules should aim to minimize the deviation between ideal demand coverage and allocated demand coverage, but also strive to increase the perceived fairness of the schedules.

When it comes to generality, the scope is, to some extent, delimited. As Medvind mainly operates in the Swedish market, the MSP is primarily concerned with legislation and work regulations within this market. In order to be sufficiently general, the solution method for generating schedules must, however, be able to solve all problems comprised by the defi-

nition of Personnel Scheduling Problems defined in Chapter 3

Input to the model is demand forecasts, information regarding the staff and a proposed set of shifts. The demand forecasts consist of information about time-specific requirements regarding staffing levels and needed competencies, while staff information involves contractual agreements, preferences and other relevant information regarding the employees. The proposed shifts are manually created shifts that are suggested as possible shifts to be used when rostering employees.

Problem areas like staffing, personnel costs, demand modeling and uncertainty are out of the scope of this master's thesis. Decisions related to staffing levels are considered fixed for the entire planning period, and personnel costs are therefore considered sunk costs. The predefined staffing has also taken into account demand modeling and uncertainty, and these problem areas are thus not included in the problem. Additionally, this master's thesis is not concerned with any events that occur during a planning period, like the sickness of employees and other sudden changes.

4.2 Description of the Medvind Scheduling Problem

The MSP addresses the problem of scheduling employees in order to satisfy a predefined demand while adhering to a set of regulations and preferences defined by governments, trade unions, managers and employees. An allocation is feasible if the demand is covered by a sufficient amount of qualified personnel, and if all of the scheduling rules and contractual agreements are satisfied. In this section, information on the main elements to consider will be presented in further detail.

4.2.1 Demand

A demand is defined by the required set of competencies, the number of employees needed to satisfy the demand, and by the time interval it occurs. The required number of employees is given as three integer values, representing the minimum, ideal and maximum coverage of the demand.

Deviation from ideal demand coverage should be penalized, as over- and understaffing is undesired. It is, however, not accepted to cover less than the minimum required demand coverage nor more than the maximum allowed demand coverage. Additionally, an employee can only cover one unit of demand for each time period and must posses the specified competence or competencies to be able to cover the corresponding demand.

4.2.2 Employees

Employees should work as close to their contracted hours as possible. An employee is allowed to deviate from the contracted hours during a week. However, the accumulated working hours over the whole planning period is not allowed to exceed the accumulated contracted hours. The deviation between the accumulated working hours and accumulated

contracted hours should be penalized. Additionally, a contract specifies information on the employee's blocked hours. This could involve personal inclinations, specific agreements or other conditions that prevent the employee from working at certain times.

Furthermore, every employee possesses a set of competencies. As mentioned in Chapter 2, these competencies will determine which demands the employee can cover. Finally, every employee will have the opportunity to state personal preferences regarding desired and undesired working times. An employee could, for instance, prefer not to work on certain evenings or to have a Friday off to extend the weekend. The schedule should aspire to take these preferences into account.

4.2.3 Shifts

A shift is defined by a starting time and a duration. There exist limits to both how short and how long a duration of a shift can be. In addition to these strict limits, there are shift durations that are more desirable than others. In other words, a 4-hours shift could be allowed even though it is preferred that shifts last at least 6-hours. Allocation of shifts of undesirable durations should, therefore, be avoided.

There is also a distinction between work shifts and off-shifts. In an off-shift, the employee is not doing any work. This implies that when allocated to an off-shift, the employees are not allowed to work or cover any units of demand. Correspondingly, an employee has to be allocated to a work shift to be able to cover demand. If an employee is allocated to a work shift, the employee has to cover demand during the entire duration of that shift. Thus, an employee cannot cover demand without being allocated to a work shift nor be allocated to work without covering any demand. Each employee can at most start one work shift each day, and cannot be assigned to overlapping shifts.

4.2.4 Required Rest

To ensure sufficient continuous rest for the employees, each employee is required to have at least one daily rest period each day and one weekly rest period each week. A daily rest period is a continuous time period with a duration that exceeds a certain threshold, for instance, eight hours, where no work is done. It is to be noted that the fulfillment of at least one daily off-period is related to employee-specific days and not calendar days. This means that the daily off-period must occur within 24 hours, but where the starting point of this interval is determined by when the employee-specific day starts. Figure 4.1 exemplifies two different requirements of daily rest. The first one involves 11 hours required rest and employee-specific day starting 04:00, while the second one involves 9 hours required rest and employee-specific day equal to a calendar day. As seen, daily rest is only satisfied in the first case.

A weekly rest period is also an off-period with a duration that exceeds the given rest threshold, but where this threshold typically is much higher compared to the daily threshold. Unlike the daily off-periods, additional weekly rest hours should be rewarded. Prolonged weekly off-periods should, therefore, be rewarded compared to shorter off-periods. In

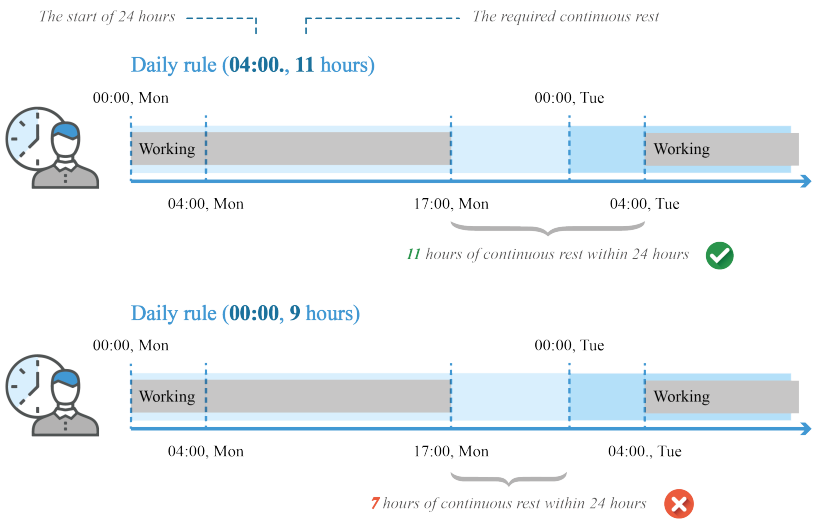

Figure 4.1: Feasible and non-feasible shift allocations with respect to required daily rest.

addition, weekly rest is related to calendar weeks. This implies that a weekly rest must be kept within a calendar week, i.e., not starting before Monday 00:00 the corresponding week, nor ending later than 24:00 the following Sunday.

4.2.5 Fairness Aspects

Schedules should aim to be as fair as possible. This implies that all of the various fairness aspects are accounted for and that the fairness is evenly distributed among the employees.

Some of the most important fairness aspects have already been discussed, like deviation from contracted hours, consideration of preferences and maximization of continuous weekly rest. In addition, there are several fairness aspects related to certain undesired shift patterns. As discussed in Chapter 2, partial weekends, isolated days, and to work more than the desired amount of consecutive days are all examples of undesired shift patterns that contribute to less attractive working conditions for the employees. These fairness aspects should be accounted for when evaluating the quality of a schedule.

4.3 Objective

The objective is to create schedules that maximize the fairness of the shift allocation among the employees and that provide a demand coverage as close to the ideal demand as possible, where the latter takes priority over the former. When maximizing the fairness, both the collective fairness level of all employees and the distribution of fairness among them should be evaluated. The schedule must also, as a minimum, adhere to all stated requirements stated in Section 4.2.

4.4 Problem Assumptions

We have made some assumptions when addressing the problem presented throughout this chapter. An overview of these assumptions is presented in Table 4.1 and are further elaborated on below.

Table 4.1: Problem assumptions

Assumption 1	Isolated planning period
Assumption 2	Preferences for time periods, not complete shifts
Assumption 3	No holidays or special events during planning period
Assumption 4	Allocations of breaks are not considered

First of all, we assume an isolated planning period is to be considered. No information is to be transferred to the next planning period, and no information is received from the previous one. In a real-life situation, it would be beneficial to receive information from the previous planning period and pass on certain information to the next period. It could, for instance, be beneficial to transfer the obtained fairness level of each employee between the planning periods, ensuring an even fairness distribution for all the planning periods, not just each individually.

Regarding employee preferences, they are given as a desire to work or *not* to work for specific time periods. There may exist situations where an employee may prefer a certain number of coherent time periods off during a day, but where this preference becomes irrelevant if not all of time periods are granted. An example is if an employee wants to attend an event from 16:00 to 20:00, but where there is no point in participating between 18:00 and 20:00. Such underlying conditions are, however, considered of negligible importance, and the granted off time from 18:00 to 20:00 is thus rewarded.

Additionally, we are assuming that the planning periods do not involve holidays, like Christmas and Easter. During such holidays, special scheduling rules may apply, and such cases are considered out of the scope of this master thesis.

Finally, we are not considering the allocation of breaks during a work shift, as it is assumed that employees themselves are able to select the most appropriate break times.

Chapter 5

Shift Design Heuristic

This chapter presents the Shift Design Heuristic, abbreviated SDH. The SDH addresses the shift design problem defined in Chapter 3 and consists of a construction heuristic that creates possible shifts to be used in the rostering stage of the MSP. In addition, an extension for shift reduction is provided as an option, aiming to reduce the search space in the rostering problem by removing candidate shifts that are not strictly necessary. The shift reduction extension is further referred to as the SR extension. The general structure of the SDH and SR extension, and how they relate to the rostering stage, is visualized in Figure 5.1. Note that the MIP model and the PALNS is presented in detail in Chapter 6 and 7, respectively.

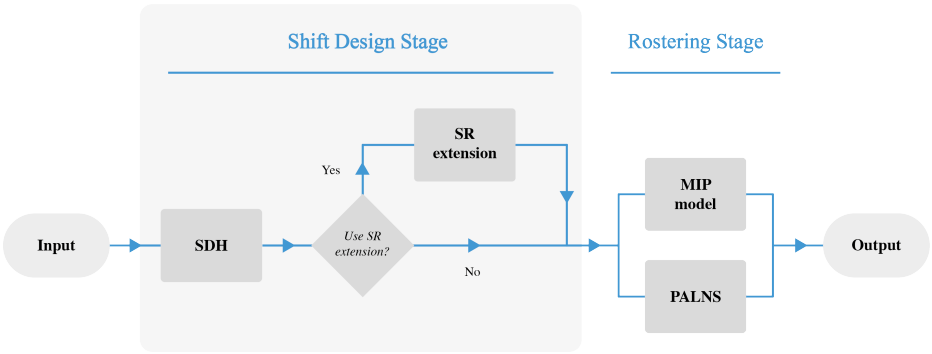

Figure 5.1: The structure of the SDH and SR extension, and how it relates to the rostering stage

This chapter starts with Section 5.1, providing a backdrop for the selected shift design approach. The chapter continues with Section 5.2, presenting the assumptions and simplifications made when addressing the shift design problem. Finally, Section 5.3 presents the proposed SDH, while Section 5.4 presents the SR extension modelled as a Mixed Integer Programming model.

5.1 Motivating the Shift Design Approach

It is not strictly necessary to solve the shift design problem to provide solutions to the MSP, but by doing so we believe that better solutions can be facilitated. The alternative to generating shifts as part of the MSP is to use manually created shifts, provided as input by Medvind. These shifts are assumed to be limiting, as they are constructed by hand and without close adaption to the specific problem that is to be solved. Additionally, the work of constructing shifts is time-consuming and difficult, especially in cases of non-cyclic and complex demand. By designing shifts as part of the MSP, we aim to automatically create shifts that are tailored to the problem at hand. The generated shifts should facilitate increased flexibility and better schedules, which is the main motivation for addressing the shift design problem.

As discussed in Chapter 1, an implicit scheduling model was developed in the preliminary work of this thesis. This implicit scheduling model is abbreviated IMP-MIP model and can be found in its entirety in Appendix A.3. The IMP-MIP model provided strong results on small problem instances. However, as indicated as a disadvantage of implicit shift models in Chapter 3, the model struggled when the problem size and complexity were increased. There was, however, one important characteristic that prevailed when interpreting the implicitly generated shifts in the obtained optimal solutions: the generated shifts had a very distinct relation to the demand. The implicitly generated shifts did, almost without exceptions, start and end in transitions between changing demand. When modeling the SDH, the idea has been to exploit this information when explicitly designing the shifts. By doing so, the nature and flexibility of the implicitly generated shifts is imitated, and the number of possible shifts could be reduced, thus avoiding the complexity issues of implicit shift design.

To facilitate flexibility, and in order to match the implicitly generated shifts, the SDH is ensured to create enough variations of shifts. Although the number of generated shifts is significantly lower than for the IMP-MIP model, the SDH could still generate a large set of shifts. In doing so, attention must be paid to the fact that most of these shifts could still be superfluous to the final schedule. This, in combination with the increased search space introduced by a large set of shifts, makes it attractive to look for ways to reduce the number of candidate shifts. This is why the SR extension is introduced, aiming to remove shifts that are not strictly necessary to provide feasible schedules. This way, the solution space is significantly reduced, but with a risk of removing shifts that could have been beneficial in order to increase scheduling flexibility. By proposing the SR extension as an extension, and not an integrated solution, the user is enabled to explore the boundaries between the concepts of scheduling flexibility and performance in order to obtain desired results.

5.2 Assumptions and Simplifications

In this section, the assumptions and simplifications made when designing the shifts will be stated. An overview of the assumptions and simplifications are presented in Table 5.1, and further discussed below.

Table 5.1: Assumptions and simplifications of the shift design model

Assumption 1	Settling with shifts assumed to be optimal
Simplification 1	Not including employee-data in the shift design process
Simplification 2	Simple handling of long and unchanging demand periods
Simplification 3	Simple handling of problems with 24 hour demand

First of all, we are not concerned with generating optimal shifts. By optimal shifts, we mean shifts that fully facilitates optimal schedules. In theory, this can only be achieved by closely coordinate the process of designing shifts and allocating employees to them, i.e., solving the personnel scheduling problem as one problem. As the results from our preliminary work manifested, this is very difficult to do for large problems. We are therefore aiming to imitate the shift structure obtained by the IMP-MIP model, and thus generate the shifts assumed to be most suitable. Our assumption is that this approach could facilitate high-quality solutions to the MSP, and additionally have a greater value compared to both implicit and predefined shifts regarding practical performance and applicability.

When it comes to simplifications, the selected shift design approach does not take into account information about the employees. Ideally, it would be advantageous to include information about employee preferences or contractual agreements, to enable the shift to better adapt to elements that might affect the performance in terms of fairness. However, this would complicate the design process, and it would be hard to do without solving the rostering problem simultaneously. This simplification thus builds on the assumption above.

Another simplification made concerns the handling of long demand periods without a change in demand. As discussed briefly in Section 5.1, our shift design approach is primarily based on transitions in demand. Put short, if demand is changing, this could trigger a relevant time to either start or end a shift. Nevertheless, in cases where the demand is not changing, for instance, if there is a constant and unchanging demand for 16 hours, no such trigger points would occur. If the maximum limit of a shift duration is 12 hours, our model has to determine reasonable ending and starting times itself in order to create shifts that cover the entire demand period. A simplified approach to handle such types of demand periods is used in the SDH, presented in more detail in Section 5.3.

The last simplification is related to how the SDH approaches problems that consist of subsequent days with 24-hour demand. For such problems, there typically exists preferred shifts structures, often related to night shifts that overlap two calendar days. It is also likely that in cases with demand for 24 hours, it might be favorable to allow shifts to start and end even when there is no change in demand. This could be favorable in order to increased flexibility or to enable the shifts to reflect more traditional structures such as day, evening and night shifts. However, the implementation of the SDH has not taken into account such considerations.

5.3 Modelling the Shift Design Heuristic

The main goal of the SDH is to facilitate high-quality schedules through the generation of favorable shifts for the rostering stage. This section aims to provide insight into the SDH and how it designs work shifts. The SDH is also responsible for creating weekly off-shifts, and this is also elaborated on. The section is concluded with a presentation on the modeling choices made.

5.3.1 Designing Work Shifts

The SDH aims to design shifts that imitate the structure of the implicitly generated shifts in the IMP-MIP model. As presented in Section 5.1, theses shifts did, almost without exception, end and start in transitions between changing demand. The idea of the SDH is to replicate this structure by generating favorable shifts that both start and end when such transitions occur.

To explain how the SDH operates, the concepts of transition times and demand periods are introduced. A transition time is the time when demand changes. Changes in demand occur both when demand is increased or decreased, or when new competencies are required. A demand period is, on the other hand, defined as the delimited intervals where demand is present. Demand periods are thus only separated by intervals where no demand exists. Note that the latter implies that in the case of two subsequent days with 24h demand, the resulting demand period would span both days. An illustration of transition times and a demand period is presented in Figure 5.2 below.

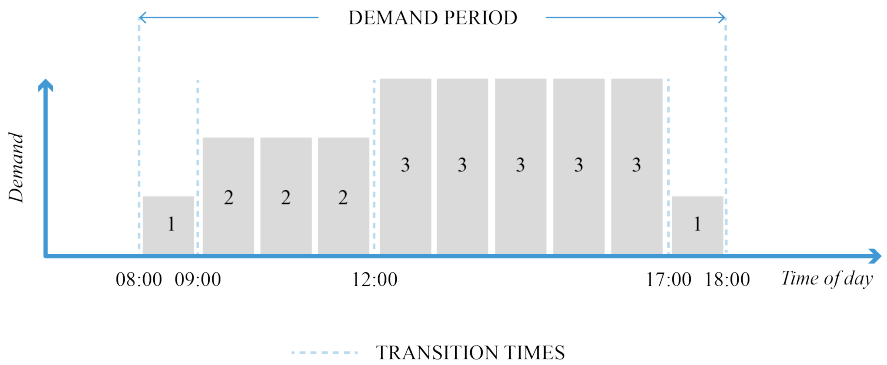

Figure 5.2: Visualization the concepts of transition time and demand period

With transition times and demand period defined, the shift design process applied in the SDH can be translated into the process of finding relevant shifts that start and end in certain transition times, and where the combination of shifts ensure to cover the entire demand period. This is the simple and intuitive idea of the SDH, and an outline of the implemented heuristic for designing work shifts is presented in Algorithm 1.

Algorithm 1 Generating Work Shifts

Input: List of demand periods for the complete planning horizon,
$\underline{V}$ and $\overline{V}$, minimum and maximum allowed shift duration

Output: Set of shifts

1: Initialize empty shift set S
2: **for** $demandPeriod$ in $demandPeriods$ **do**
3: **while** $demandPeriod$ is not empty **do**
4: $demandCovered$ = False
5: $interval$ = $pop(demandPeriod)$
6: **if** $getDuration(interval) < \overline{V}$ and $demandPeriod$ is empty **then**
7: Add shift($getStart(interval), getDuration(interval)$) to S
8: Continue from line (3)
9: **for** transition time t in $demandPeriod$ **do**
10: $duration$ = t - $getStart(interval)$
11: **if** $\underline{V} < duration < \overline{V}$ **then**
12: Add shift($getStart(interval)$, $duration$) to S
13: $demandCovered$ = True
14: **else if** $duration > \overline{V}$ **and not** $demandCovered$ **then**
15: Add shifts created by $CreateShiftsForLongDemandIntervals()$
16: $demandCovered$ = True
17: **return** S

As seen in Algorithm 1, the list of demand periods for the entire planning horizon is provided as input. For each demand period, the algorithm is generating all feasible shifts that are possible to allocate between the various transition times. Note how the SDH handles demand periods with duration shorter than $\underline{V}$. Such special cases occur if an organization wants to hold short meetings in periods where there is no other demand. Shift shorter than $\underline{V}$ is allowed for such cases. This is handled in line (6) to (8).

For the demand structure presented in Figure 5.2, the SDH would produce the shifts presented in Figure 5.3. Note that the shifts can be selected in various ways in order to satisfy the demand. One schedule could involve Shift 1, Shift 4 and Shift 5, while another could involve Shift 2, Shift 3 and Shift 5. The combination of Shift 2, Shift 4 and Shift 5 constitutes a third possible schedule. These scheduling alternatives illustrate the concept of scheduling flexibility, where the scheduling manager can choose from several feasible shift

combinations. Note also that for all the proposed schedules, only three out of six shifts are used. By facilitating scheduling flexibility, more shifts than what is needed are generated, and the rostering process thus becomes more complicated. This demonstrates the contradictions between scheduling flexibility and performance.

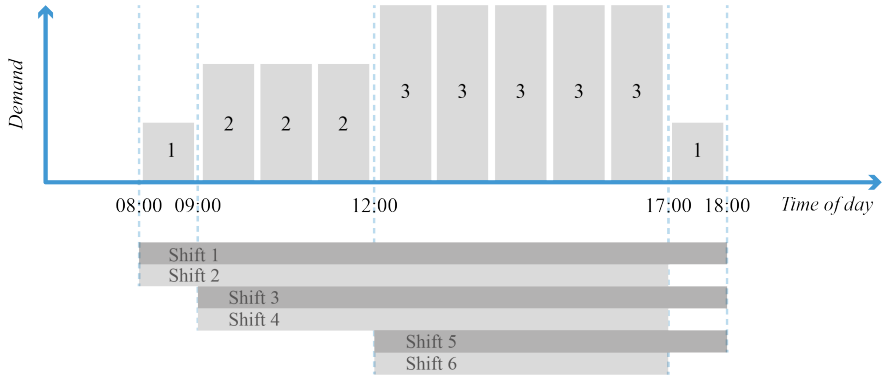

Figure 5.3: Example of the structure of the shifts generated by the SDH

As discussed in Section 5.2, for demand intervals with a duration exceeding the maximum limit of allowed shift duration, the SDH has to select reasonable ending and starting times for the shifts. This is handled by the function *CreateShiftsForLongDemandIntervals()*. This function takes the long demand interval, as well as the minimum and maximum required shift duration as input. The demand interval is then split into two intervals of equal length, and two shifts corresponding to each interval are created. Additionally, the original demand interval is split into three demand intervals of equal length. If all of these intervals are of a duration greater than the minimum required shift length, three additional shifts corresponding to these three demand intervals are created. The latter is done in order to improve flexibility and provide a more diverse set of shifts. An example of the resulting output-shifts of the *CreateShiftsForLongDemandIntervals()* is visualized in Figure 5.4.

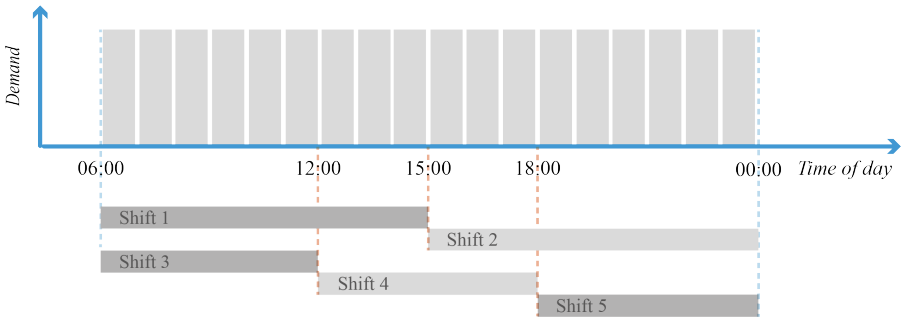

Figure 5.4: Example of how the SDH handles long periods with unchanging demand

5.3.2 Designing Weekly Off-Shifts

The weekly off-shifts are off-shifts that is to ensure that the employees are allocated a sufficient amount of continuous rest each week. One fairness aspect considered in Chapter 2 is the maximization of weekly rest. Since employees should receive as long weekly rest as possible, the SDH has to design off-shifts that facilitate long rest periods and not only off-shifts that satisfy the minimum weekly rest requirement. Additionally, the weekly off-shifts must start and end within the calendar week it applies to and are thus not allowed to overlap between calendar weeks.

The implementation of how the SDH generates the weekly off-shifts is presented in Algorithm 2. Note that the generation of weekly off-shifts is based on the already generated work shifts, provided as an input and sorted to the week the shift applies. The idea of the weekly off-shift generation is to exploit the fact that there is no point in starting a weekly off-shift for other times than when a work shift is ended. If a weekly off-shift is not starting when a work shift is ended, this will imply that the rest hours from when the previous work shift ended is not utilized. Based on the same underlying idea, the algorithm is only creating weekly off-shifts that end when a subsequent work shift is started.

Algorithm 2 Generating Weekly Off-Shifts

Input: S^{week}, set of works shifts belonging to each week,
$\underline{V}^R$ and $\overline{V}^R$, minimum and maximum weekly rest duration

Output: List of weekly rest shifts

1: Initialize empty weekly off-shift set, $R = [\,]$
2: **for** week w in $weeks$ **do**
3: If necessary, add dummy shifts to S_w^{week}
4: **while** S_w^{week} is not empty **do**
5: $shift$ = $pop(S_w^{week})$
6: **for** shift s in S_w^{week} **do**
7: **if** time $getEnd(s) \geq$ time $getEnd(w)$ **then**
8: Continue from line (4)
9: $duration$ = $getStart(s) - getEnd(shift)$
10: **if** $\underline{V}^R \leq duration \leq \overline{V}^R$ **then**
11: Add offShift($getEnd(shift)$, $duration$)
12: **return** R

Note that in line 3, dummy shifts are added if necessary. This is done in order to utilize the first and last hours of a week. In cases where there are no shift ending at the first time

in a week and no shift starting in the last time in a week, corresponding dummy shifts are introduced in order to enable Algorithm 2 to search for weekly off-shifts utilizing as many hours as possible.

The input parameters $\underline{V}^R$ and $\overline{V}^R$ represents the limits for weekly off-shift durations that the Algorithm 2 is enabled to generate. The $\underline{V}^R$ equals the minimum required amount of weekly rest and is stated by the user. The $\overline{V}^R$, on the other hand, is a little more complicated. Briefly explained, an upper limit is set for the maximum number of hours that are rewarded. Since the weekly off-shift algorithm is searching for shifts that fit exactly between work shifts, the algorithm could fail in generating favorable off-shifts if the $\overline{V}^R$ was set equal to the maximum rewarded weekly off-shift duration. This scenario is visualized in Figure 5.5 below, with the required number of rest hours set to 36 hours and maximum rewarded number of hours set to 72 hours.

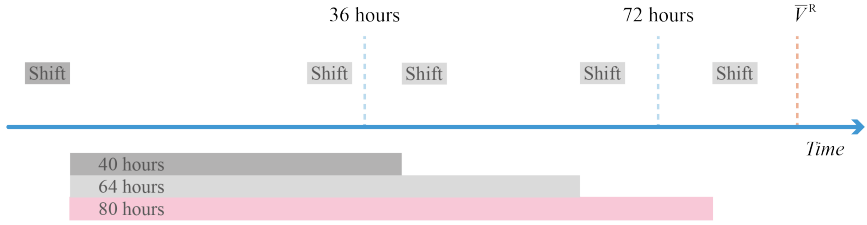

Figure 5.5: Visualization of why the $\overline{V}^R$ parameter is introduced

As seen, when the algorithm is only enabled to look for off-shifts with durations in the interval of 36 and 72 hours, the algorithm fails to generate the off-shift with 80 hours rest - colored red in Figure 5.5. This limits the opportunity of maximizing weekly rest, as the 80-hour off-shift could be allocated if it was generated. By enabling Algorithm 2 to increase the upper limit for the shift length to $\overline{V}^R$, we see that the 80-hour shift will be generated.

5.3.3 Modelling Choices and Insight

This section elaborates on key modeling choices and aims to provide increased insight into the approach when developing the SDH. Some areas of improvement are also identified and will be addressed and discussed. An overview of the content in this section is provided in Table 5.2 below.

Table 5.2: Prominent modelling choices for the SDH

Choice 1	Using transition times as basis for creating shifts
Choice 2	Creating time-specific shifts

As mentioned, the SDH is developed based on the indication that the implicitly generated shifts from the IMP-MIP model commonly start and end in transition times. This is an intuitive and logical shift structure. When demand increases, this constitutes a natural time for an employee to start a shift. The same is true when demand decreases, only that this often represents a suitable time to end a shift. There is, however, one drawback related to our strict attention towards transition times, which becomes evident when evaluating the generation of weekly off-shifts.

As presented in Subsection 5.3.2, the generation of weekly off-shifts is based on finding relevant off-periods between work shifts. As seen in Figure 5.5, the $\overline{V}^R$ is introduced in order to enable off-shifts that can contribute to maximizing the reward of weekly rest. This approach is selected as the algorithm is searching for an interval between shifts, thus between transition times. This is, however, both an inefficient and unreliable approach, as it makes the algorithm sensitive to the value of the $\overline{V}^R$. This will especially be the case in scenarios with subsequent days without demand, or if the $\overline{V}^R$ parameter is set too low such that desirable shifts are still not generated. It would thus be highly advantageous to detach from the strict relation to transition times. By creating an off-shift of a duration equal to the maximal rewarded weekly rest, if no such off-shift was found and regardless of whether the off-shift is ended a transition time or not, the aforementioned inefficiencies and unreliability would be avoided. In this way, the algorithm would have ensured that off-shift enabling the maximum reward of weekly rest always was generated, and the $\overline{V}^R$ parameter would be unnecessary.

The second modeling choice addressed is the use of time-specific shifts. By time-specific shifts, we mean that shifts are associated with the specific time it starts. To exemplify, imagine a shift starting at 08:00 and ending at 16:00. This shift could be used on a Monday and be repeated the next day. Such shifts are traditionally viewed as the same shift, i.e., a shift starting 08:00 and ending at 16:00. In the SDH, these shifts are, however, treated as individual shifts as 08:00 on Monday is regarded as a specific time different from the time 08:00 the following Tuesday. This is a modeling choice made to preserve the generality, enabling an efficient approach to problems that potentially have a unique demand for each day throughout the planning period. In this way, shifts can be specifically tailored to each demand day. Additionally, by generating time-specific shifts, the shifts can easily be related to different periods in the planning horizon directly, enabling a more efficient search for relevant shifts in the rostering stage.

5.4 Shift Reduction Extension

In this section, the shift reduction extension, denoted SR extension, is presented in detail. The purpose of the SR extension is to select the shifts that enable demand coverage as close to ideal demand coverage as possible while at the same time minimizing the number of shifts used. This is done in order to reduce the search space for the solution methods in the rostering stage. The SR extension is modeled as a MIP model, and this section starts with an introduction to the definitions used when modeling the extension. The objective

of the model is then formulated, followed by an elaboration of the restrictions imposed on the SR extension.

5.4.1 Definitions

The following sets, parameters and variables are used when modeling the SR extension of the SDH. Sets are defined in calligraphic capital letters, while parameters are defined using Latin uppercase letters. Variables are defined by both lowercase Latin letters and lowercase Greek letters, where Latin letters indicate decision variable, and Greek letters indicate helper variables. The decision variables affect the feasibility of the problem directly, while the helper variables are introduced to relate to particular objectives. Subscript indicate indices, while superscript indicate specific meanings for some sets, parameters and variables. Over- and underlines are used to indicate upper and lower limits.

Sets

$\mathcal{T}$: Set of time periods with demand in planning horizon

$\mathcal{S}$: Set of shifts

$\mathcal{S}_t^T$: Set of shifts overlapping time t

$\mathcal{S}^L$: Set of shifts with duration longer than desired duration

$\mathcal{S}^S$: Set of shifts with duration shorter than desired duration

$\mathcal{C}$: Set of competencies

The user is enabled to specify the minimum and maximum allowed duration for a shift. However, even though a shift has a valid duration, the shift duration might not be desired. Therefore, the user is also enabled to specify upper and lower limits for desired shift duration, and the sets $\mathcal{S}^L$ and $\mathcal{S}^S$ contain shifts that do not meet these desired limits.

Parameters

$\underline{D}_{ct}$: Minimum coverage of demand for competency type c at time t

D_{ct} : Ideal coverage of demand for competency type c at time t

$\overline{D}_{ct}$: Maximum coverage of demand for competency type c at time t

$\overline{V}$: Maximal desired shift duration

$\underline{V}$: Minimal desired shift duration

V_s : Duration of shift s

M : Big M for mapping use of shift

W^{D+} : Weight for positive deviation between actual and ideal demand coverage

W^{D-} : Weight for negative deviation between actual and ideal demand coverage

W^L : Weight for use of shifts with durations exceeding desired limit

W^S : Weight for use of shifts with durations shorter than desired limit

Variables

y_s = $\begin{cases} 1, & \text{if shift } s \text{ is used} \\ 0, & \text{otherwise} \end{cases}$

x_s = Number of times shift s is used

μ_t = difference between covered demand and minimum demand at time t

$\delta_t^+,\ \delta_t^-$ = excess/deficit on coverage of demand at time t relative to ideal demand

5.4.2 Objective

The objective function of the SR extension is presented in Equation (5.1). The main objective of the SR extension is to reduce the number of possible shifts, thus contribute to reducing the search space in the subsequent rostering stage. This is represented by the term (5.1a) in the objective function below. At the same time, it is important that the SR extension does not reduce the number of shifts at the expense of high-quality schedules. An important aspect of the MSP is demand coverage. It is therefore desirable that the SR-extension permits as tight demand coverage as possible. This is ensured by the term (5.1b) in the objective function. Note that the W^{D+} and W^{D-} weights enable the user to weight under-coverage and over-coverage differently. The last term (5.1c) penalizes the the use of shifts with undesired shift durations. In some cases, it can be sensible, or even required, to allow for shorter or longer shifts than what is normally desired. This can be crucial to achieve a closer demand coverage or to enable satisfactory coverage of special demand like peak demands. Such shifts are, however, undesirable by employees and should be avoided when possible. The use of shifts with undesirable durations is thus penalized according to the magnitude of the violation. Note that the W^L and W^S weights enable the user to weight the use of too long and too short shifts differently.

$$\min z = \sum_{s \in \mathcal{S}} y_s \tag{5.1a}$$

$$+ W^D \cdot \sum_{t \in \mathcal{T}} (\delta_t^+ + \delta_t^-) \tag{5.1b}$$

$$+ W^L \cdot \sum_{s \in \mathcal{S}^L} (V_s - \overline{V}) \cdot y_s + W^S \cdot \sum_{s \in \mathcal{S}^S} (\underline{V} - V_s) \cdot y_s \tag{5.1c}$$

5.4.3 Constraints

Constraints (5.2) ensure that the minimum demand is covered for all periods, in addition to storing excess demand coverage in the μ-variable, relative to the minimum demand requirement. Constraints (5.3) ensures that excess demand stays between the minimum and maximum demand coverage limits. Finally, constraints (5.4) store the deviation between ideal demand coverage and actual demand coverage in the respective δ-variables. Any deviations are penalized in the objective function, and thus it is no need to limit the upper bounds of these variables.

$$\sum_{s \in \mathcal{S}_t^T} x_s = \sum_{c \in \mathcal{C}} \underline{D}_{ct} + \mu_t \qquad t \in \mathcal{T} \tag{5.2}$$

$$\mu_t \leq \sum_{c \in \mathcal{C}} (\overline{D}_{ct} - \underline{D}_{ct}) \qquad t \in \mathcal{T} \tag{5.3}$$

$$\mu_t + \sum_{c \in \mathcal{C}} (\underline{D}_{ct} - D_{ct}) = \delta_t^+ - \delta_t^- \qquad t \in \mathcal{T} \tag{5.4}$$

Constraints (5.5) maps the relation between the x- and y-variables. The big M ensures that the y-variable is only set to zero if the s shift is not used throughout the planning horizon.

$$x_s \leq M \cdot y_s \qquad s \in \mathcal{S} \tag{5.5}$$

Constraints (5.6) to (5.9) constitutes the final constraints of the SR extension. They all impose restrictions on the variables used in the model.

$$y_s \in \{0, 1\} \qquad s \in \mathcal{S} \tag{5.6}$$

$$x_s \geq \mathbb{Z}^+ \qquad s \in \mathcal{S} \tag{5.7}$$

$$\mu_t \geq \mathbb{Z}^+ \qquad t \in \mathcal{T} \tag{5.8}$$

$$\delta_t^+,\ \delta_t^- \geq \mathbb{Z}^+ \qquad t \in \mathcal{T} \tag{5.9}$$

Chapter 6

Mathematical Model

In this chapter, a mathematical model formulated as a linear Mixed Integer Programming model is presented in order to solve the rostering stage of the MSP. The implemented mathematical model is hereby referred to as the MIP model. The sets, parameters and variables used to formulate the MIP model is first presented in Section 6.1. The objective function is then formulated in Section 6.2, while Section 6.3 presents how the rules of the MSP are modelled as constraints. The chapter is concluded by Section 6.4, aiming to discuss and provide increased insight into relevant modeling choices made.

6.1 Definitions

In this section, the sets, parameters and variables used to model the MSP are defined. Sets are defined in calligraphic capital letters, while parameters are defined using Latin uppercase letters. Variables are defined by both lowercase Latin letters and lowercase Greek letters, where Latin letters indicates decision variable and Greek letters indicates helper variables. Subscript indicate indices, while superscript indicates specific meanings for some sets, parameters and variables.

6.1.1 Sets

$\mathcal{C}$: Set of competencies

$\mathcal{E}$: Set of employees

$\mathcal{E}_c^C$: Set of employees with competency c

$\mathcal{I}$: Set of days in planning horizon

$\mathcal{I}^{SAT}$: Set of Saturdays, indicating start of weekend

$\mathcal{J}$: Set of weeks in planning horizon

$\mathcal{T}$: Set of time periods with demand in the planning horizon

$\mathcal{T}_r^R$	:	Set of time periods with demand in weekly off-shift r
$\mathcal{S}$	:	Set of work shifts
$\mathcal{S}_i^I$	:	Set of possible work shifts at day i
$\mathcal{S}_t^T$	:	Set of work shifts overlapping time t
$\mathcal{S}_e^{INV}$	:	Set of invalid work shifts for employee e
$\mathcal{S}_{es}^{DRC}$	:	Set of shifts that violates daily rest for employee e if shift s is worked
$\mathcal{S}_{es}^{DRS}$	:	Set of shifts that violates daily rest for employee e if two or more of them are worked in addition to shift s
$\mathcal{R}$	:	Set of weekly off-shifts
$\mathcal{R}_j^J$	:	Set of weekly off-shifts in week j

The set $\mathcal{S}_i^I$ contains shifts that belonging to day i. It is to be noted that the time defining what day a shift belongs to is not restricted to midnight, but given as input by the user. This implies that a shift starting at 22:00 could be defined as a night shift belonging to the next calendar day. The set $\mathcal{S}_e^{INV}$ contains shifts that are invalid to employee e. Invalid shifts constitutes shifts that are overlapping blocked hours or that violate daily rest, and that therefore cannot be worked by the employee. The sets $\mathcal{S}_{es}^{DRC}$ and $\mathcal{S}_{es}^{DRS}$, on the other hand, contain invalid shift combinations and shift sequences, respectively, that violate daily rest. Examples on invalid shift combinations and invalid shift sequences are visualized in Figure 6.1.

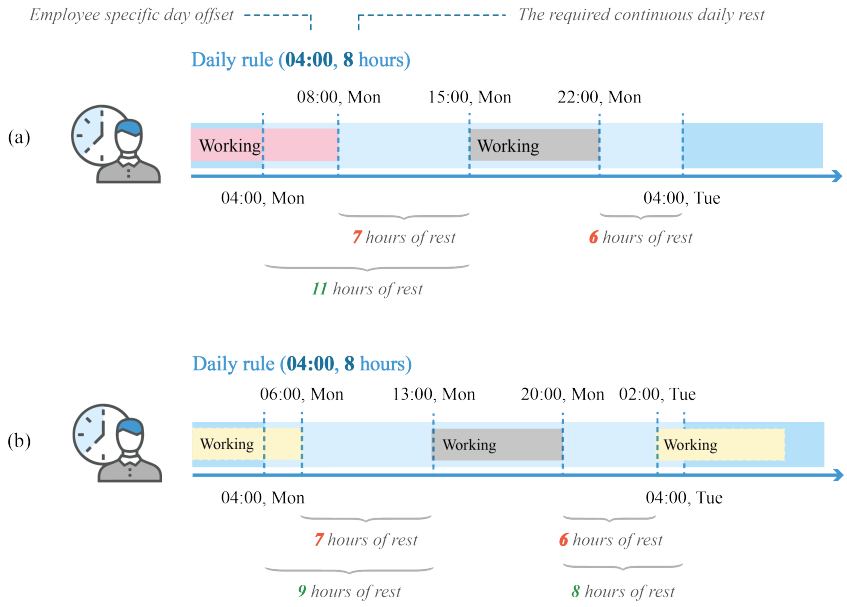

Figure 6.1: (a) Example on invalid shift combination and (b) example on invalid shifts sequences.

Figure 6.1 (a) illustrates that if the grey shift is worked, the red shift cannot be worked due to daily rest. The grey and red shifts therefore constitutes an invalid shift combination. Figure 6.1 (b), on the other hand, illustrates an invalid shift sequence. If both the yellow shifts are worked in addition to the grey shift, daily rest is violated. But if, however, only one of the yellow shifts is worked, daily rest is satisfied. The combination of the two yellow shifts and the grey shift thus constitutes a invalid shift sequence.

6.1.2 Limit Parameters

The limit parameters impose limits on the constraints. D is used for demand parameters. B is used for contracted hours, and L is used for limits on consecutive days. Over- and underlines are used to indicate if a parameter represents an upper or lower limit.

- $\underline{D}_{ct}$: Minimum coverage of demand for competency type c at time t
- D_{ct} : Ideal coverage of demand for competency type c at time t
- $\overline{D}_{ct}$: Maximum allowed coverage of demand for competency type c at time t
- B_e : Amount of weekly contracted time periods for employee e
- L^{CD} : Desired limit of consecutive working days

6.1.3 Weighting Parameters

The weighting parameters represent intentional weights. These weights are set by the user and stipulate how the MIP model should prioritize the different scheduling aspects affecting the objective score of a schedule.

- W^{D+} : Weight for positive deviation between actual and ideal demand coverage
- W^{D-} : Weight for negative deviation between actual and ideal demand coverage
- W^{F} : Weight for fairness score of least favored employee
- W^{R} : Weight for maximum continuous weekly rest
- W^{B} : Weight for deviation between worked and contracted hours
- W^{PW} : Weight for partial weekends
- W^{IW} : Weight for isolated working days
- W^{IO} : Weight for isolated off-days
- W^{CD} : Weight for consecutive days
- W^{P} : Weight for granting preferences

6.1.4 Adjustment Parameters

The adjustment parameters are used to adjust some of the weights in order to preserve their intended purpose. As the magnitude of certain scheduling aspects depends on problem-specific characteristics, like the time step or number of employees, the MIP model could consequently yield different results depending on theses problem characteristics. This is not desirable, and the adjustment parameters are thus introduced. How the adjustment parameters are calculated is discussed in greater detail in Chapter 8.

A^D : Adjusting factor for weighting of demand deviation

A^F : Adjusting factor for weighting of fairness score of least favoured employee

A_e^B : Adjusting factor for weighting of deviation in contracted hours for employee e

A^P : Adjusting factor for weighting of preferences

6.1.5 General Parameters

The general parameters are parameters that do not fit into any of the previous categories. V is used for parameters related to shift duration. P are used for preference parameters.

V_r^R : Duration of weekly off-shift r

$\overline{V}^R$: Maximum rewarded amount of weekly rest

P_{et} : Preference for working the time period t for employee e. Negative value indicating preference for not working, positive value indicating preference for work and 0 indicating indifference

6.1.6 Variables

$$x_{es} = \begin{cases} 1, & \text{if employee } e \text{ works shift } s \\ 0, & \text{otherwise} \end{cases}$$

$$y_{cet} = \begin{cases} 1, & \text{if employee } e \text{ covers one unit of demand for competence } c \text{ at time } t \\ 0, & \text{otherwise} \end{cases}$$

$$w_{er} = \begin{cases} 1, & \text{if employee } e \text{ takes the weekly off-shift } r \\ 0, & \text{otherwise} \end{cases}$$

f_e = variable storing the fairness score of employee e

g = variable storing the lowest fairness score among the employees

$$\gamma_{ei} = \begin{cases} 1, & \text{if employee } e \text{ works day } i \\ 0, & \text{otherwise} \end{cases}$$

$$\rho_{ei}^{SAT}, \rho_{ei}^{SUN} = \begin{cases} 1, & \text{if employee } e \text{ works Saturday/Sunday and not Sunday/Saturday on day } i \\ 0, & \text{otherwise} \end{cases}$$

$$\sigma_{ei} = \begin{cases} 1, & \text{if employee } e \text{ has an isolated work day on day } i \\ 0, & \text{otherwise} \end{cases}$$

$$\phi_{ei} = \begin{cases} 1, & \text{if employee } e \text{ has an isolated off-day on day } i \\ 0, & \text{otherwise} \end{cases}$$

$$\pi_{ei} = \begin{cases} 1, & \text{if employee } e \text{ on day } i \text{ starts a sequence of consecutive working days exceeding desired limit} \\ 0, & \text{otherwise} \end{cases}$$

δ_{ct}^{+}, δ_{ct}^{-} = excess/deficit on coverage of demand for competency c at time t according to the ideal demand

λ_e = deviation in worked and contracted time periods for employee e

μ_{ct} = difference between covered demand and minimum demand for competency type c and shift type s at time t

6.2 Objective Function

The objective function (6.2) aims to maximize the fairness of a schedule, while minimizing the deviation between the actual demand coverage and the ideal demand coverage.

The fairness of a solution is related to how well the solution adhere to the various fairness aspects defined by employees, workplace or labor unions. Each employee is assigned an individual fairness score, f_e, indicating the achieved fairness level. To ensure a fair schedule, where the fairness is evenly distributed among the employees, the objective function includes two dimensions of fairness. The first dimension is the aggregated fairness score across the entire staff, represented by term (6.1a). The second dimension is the individual fairness score of the least favoured employee, represented by the term (6.1b). This term is

introduced to facilitate evenly distributed fairness scores among the employees, to prevent that increased fairness is achieved on the expense of individual employees. The parameter W^F adjusts the relative weight between the fairness of the least favored employee and the collective fairness of the entire staff, while the adjusting factor A^F scale the contribution of the least favoured employee in order to compensate the problem dependent magnitude of the accumulated fairness term.

The final term (6.1c) is penalizing deviation between the allocated coverage of demand and the stated ideal coverage of demand. Note that the weights W^{D+} and W^{D-} make it possible to weight over- and understaffing differently. The adjusting factor A^D ensures that the demand deviation term is represented in complete hours of demand deviation.

$$\max z = \sum_{e \in \mathcal{E}} f_e \tag{6.1a}$$

$$+ W^F \cdot A^F \cdot g \tag{6.1b}$$

$$- A^D \cdot \sum_{c \in \mathcal{C}} \sum_{t \in \mathcal{T}} (W^{D+} \cdot \delta_{ct}^{+} \; + \; W^{D-} \cdot \delta_{ct}^{-}) \tag{6.1c}$$

6.3 Constraints

In this section, all constraints included in the mathematical model is presented and explained in further detail. The constraints are grouped into categories based on their functionality.

Demand Coverage

As stated in Chapter 4, for each time period there is one or more demands that are to be covered. The demand is given as three values, stating the minimum required coverage of the demand, the ideal coverage and the maximum allowed coverage of demand for each time period. Constraints (6.2) make sure that the minimum coverage of demand is met at each time period for all competencies. Additionally it stores excess staffing in the μ_{ct}-variables, relative to the minimum required demand coverage. Constraints (6.3) ensure that the excess demand stays between the minimum and maximum demand coverage limits, as otherwise would indicate a violation of the minimum and maximum coverage restrictions. Constraints (6.4) calculate the deviation between ideal demand and actual covered demand, and stores the surplus or deficit in the respective δ_{ct}^{+}- and δ_{ct}^{-}-variables.

$$\sum_{e \in \mathcal{E}_c^C} y_{cet} = \underline{D}_{ct} + \mu_{ct} \qquad c \in \mathcal{C},\ t \in \mathcal{T} \tag{6.2}$$

$$\mu_{ct} \leq \overline{D}_{ct} - \underline{D}_{ct} \qquad c \in \mathcal{C},\ t \in \mathcal{T} \tag{6.3}$$

$$\mu_{ct} + \underline{D}_{ct} - D_{ct} = \delta_{ct}^{+} - \delta_{ct}^{-} \qquad c \in \mathcal{C},\ t \in \mathcal{T} \tag{6.4}$$

Work Allocation

An employee is only allowed to work at most one shift per day. This is taken care of by constraints (6.5), that in addition assigns the correct value to the binary helper variables γ_{ei}, indicating whether an employee is working day i or not. Additionally, if an employee is assigned to a shift, the employee need to cover demand throughout that shift. An employee can thus not be allocated to a shift without doing any work. In the same way, if an employee is set to cover a unit of demand the employee will need to be allocated to a shift that spans the time period where the demand is covered. This is all handled by constraints (6.6). Another restriction is that an employee can cover maximum one unit of demand for each time period. This is ensured by constraints (6.7). Note that the combination of (6.6) and (6.7) also ensure no overlapping shifts.

$$\sum_{s \in \mathcal{S}_i^I} x_{es} = \gamma_{ei} \qquad e \in \mathcal{E},\ i \in \mathcal{I} \tag{6.5}$$

$$\sum_{s \in \mathcal{S}_t^T} x_{es} = \sum_{c \in \mathcal{C}} y_{cet} \qquad e \in \mathcal{E},\ t \in \mathcal{T} \tag{6.6}$$

$$\sum_{c \in \mathcal{C}} y_{cet} \leq 1 \qquad e \in \mathcal{E},\ t \in \mathcal{T} \tag{6.7}$$

Rest

Constraints (6.8) and (6.9) ensure that the rule of daily rest is satisfied. Constraints (6.8) make sure that invalid shifts combinations are not worked. Constraints (6.9), on the other hand, make sure that invalid shift sequences are avoided. A shift sequence implies that an employee can work a certain shift on day one, and a certain shift on day two, but that the combination of these two shifts makes it impossible to work a certain shift on day three without violating the daily rest. Thus, only one shift in the set $\mathcal{S}_{es}^{DRS}$ could be worked if shift s is worked. Note that both $\mathcal{S}_{es}^{DRC}$ and $\mathcal{S}_{es}^{DRS}$ only contains shifts that belong to either the previous or next day, relatively to the day shift s belong to. This, in combination with only allowing one shift each day, ensures that at maximum of two shifts in each of these sets can be worked. This fact makes both (6.8) and (6.9) valid, even if shift s is not worked.

Constraints (6.10) make sure that each employee is allocated to one weekly off-shift each week. The constraints (6.11), on the other hand, ensure that the employee is not covering any demand during the weekly off-shift, and thus do not work during that off-shift.

$$2 \cdot x_{es} + \sum_{s' \in \mathcal{S}_{es}^{DRC}} x_{es'} \leq 2 \qquad e \in \mathcal{E},\ s \in \mathcal{S} \mid \mathcal{S}_{es}^{DRC} \neq \emptyset \tag{6.8}$$

$$x_{es} + \sum_{s' \in \mathcal{S}_{es}^{DRS}} x_{es'} \leq 2 \qquad e \in \mathcal{E},\ s \in \mathcal{S} \mid \mathcal{S}_{es}^{DRS} \neq \emptyset \tag{6.9}$$

$$\sum_{r \in \mathcal{R}_j^J} w_{er} = 1 \qquad e \in \mathcal{E},\ j \in \mathcal{J} \tag{6.10}$$

$$|\mathcal{T}_r^R| \cdot w_{er} \leq \sum_{t \in \mathcal{T}_r^R} (1 - \sum_{c \in \mathcal{C}} y_{cet}) \qquad e \in \mathcal{E},\ r \in \mathcal{R} \tag{6.11}$$

Contracted Hours

Employees are allowed to work less, but not more than the aggregated sum of contracted hours for the entire planning period. Constraints (6.12) enforce this, and store the deviation between worked and contracted hours in the non-negative λ_e-variables.

$$\sum_{c \in \mathcal{C}} \sum_{t \in \mathcal{T}} y_{cet} + \lambda_e = |\mathcal{J}| \cdot B_e \qquad e \in \mathcal{E}, \tag{6.12}$$

Partial Weekends

Constraints (6.13) store the number of partial weekends where an employee works only Saturday in ρ_{ei}^{SAT}, and the partial weekends where an employee works only Sunday in ρ_{ei}^{SUN}. Since both these variables are penalized in the fairness score, they will be set to zero if an employee works either a full weekend or have the entire weekend off.

$$\gamma_{ei} - \gamma_{e(i+1)} = \rho_{ei}^{SAT} - \rho_{e(i+1)}^{SUN} \qquad e \in \mathcal{E},\ i \in \mathcal{I}^{SAT} \tag{6.13}$$

Isolated Days

It is neither desirable to have an isolated working day or an isolated day off. Isolated working days are managed by constraints (6.14), while isolated off-days are managed by constraints (6.15). As both of these patterns are penalized in the fairness score, it is not necessary to explicitly restrict the upper bound for σ_{ei} and ϕ_{ei}.

$$-\gamma_{e(i-1)} + \gamma_{ei} - \gamma_{e(i+1)} \leq \sigma_{ei} \qquad e \in \mathcal{E},\ i \in \{2, ..., |\mathcal{I}| - 1\} \tag{6.14}$$

$$\gamma_{e(i-1)} - \gamma_{ei} + \gamma_{e(i+1)} - 1 \leq \phi_{ei} \qquad e \in \mathcal{E},\ i \in \{2, ..., |\mathcal{I}| - 1\} \tag{6.15}$$

Consecutive Working Days

To reduce the workload experienced by the employees, there is a desired limit on the number of consecutive working days. Constraints (6.16) track if the desired maximum amount of consecutive working days are violated. As π_{ei} is penalized in the fairness score, there is no need to enforce a upper bound on these variables.

$$\sum_{i'=i}^{i'+L^{CD}} \gamma_{ei'} - L^{CD} \leq \pi_{ei} \qquad e \in \mathcal{E},\ i \in \{1, ..., |\mathcal{I}| - L^{CD}\} \tag{6.16}$$

Desired Working Times

Each employee has the opportunity to state the times when they prefer to work or when they prefer *not* to work. If an employee have a preference for working certain time periods, the corresponding parameter P_{et} will be positive. Conversely, if the employee has a preference for not working at a certain time, the corresponding P_{et} will be negative. The P_{et} is defined for the interval [-1, 1] so that different values can be used as weighting among mutual preferences, i.e the value 0.8 could indicate a stronger preference for work than the value 0.2, and so on. P_{et} is used directly in the fairness constraints (6.17) to reward allocation of work to desired times periods and penalizing allocation of work on undesired time periods.

The P_{et}-parameters are normalized based on the number of preferences stated by the employee. This means that if an employee has more preferences, each preference is weighted less compared to the preferences of an employee stating fewer preferences. In this way, employees can communicate a stronger desire for certain preferences by only stating the most preferred ones.

Blocked Hours

Blocked hours indicates time periods during a planning period where an employee cannot be allocated to work, and the various reasons for blocked hours are discussed in Chapter 4. Blocked hours are handled by making shifts overlapping these blocked hours invalid to the respective employee, and are thus ensured through constraints (6.19) presented later.

Fairness

As mentioned earlier, each employee is assigned an individual fairness score f_e, where a higher score indicates a more fair solution. The fairness score is made up of the terms

(6.17a) - (6.17g), all of which represents a certain fairness aspect.

It is preferable that the weekly rest is as long as possible. The term (6.17a) takes this aspect into account, by rewarding weekly off-shifts with the longest durations. Note however that $\overline{V}^R$ is the maximum weekly rest duration that is rewarded. An employee should in addition work as close to the contracted hours as possible, and the term (6.17b) ensure that deviation from contracted hours is penalized in the solution. The A_e^B parameters scale the λ_e variable to represent hourly deviation, in addition to scaling the W^B weight to account for the amount of contracted hours an employee has. This is desirable, as four hours deviation should be penalized harder for a part-time employee with 15 contracted hours, than for an employee with 40 contracted hours. Term (6.17c) penalizes allocation of partial weekends, while the terms (6.17d) and (6.17e) penalizes occurrences of isolated working days and isolated off-days, respectively. If the desired limit of consecutive working days is violated, term (6.17f) ensures that this affects the fairness score negatively. Term (6.17g) account for employee preferences, as allocations of work to desired time periods are rewarded and allocations of work to undesired time periods are penalized. As preferences are given per time period, the adjusting parameter A^P scale the contribution of preferences to represent hourly granted or rejected preferences. In addition, the A^P ensures that the values of the preference values P_{et} is scaled based on the length of the planning horizon. As each employee is able to state a number of preferences each week, this contributes to the fact that a longer planning horizon lead to more preferences stated. Since the preference values are normalized based on the total amount of preferences for the entire planning horizon, the length of the planning horizon must be accounted for.

The individual fairness score for the least favored employee is obtained through constraints (6.18), that limits the upper bound of g to be the minimum of all fairness scores f_e. As the term is reward in the objective function there is no need to limit the lower bound of g.

$$
\begin{aligned}
f_e = \quad & W^R \cdot \sum_{r \in \mathcal{R}} \min(V_r^R, \overline{V}^R) \cdot w_{er} && && \text{(6.17a)} \\
& - W^B \cdot A_e^B \cdot \lambda_e && && \text{(6.17b)} \\
& - W^{PW} \cdot \sum_{i \in \mathcal{I}^{SAT}} (\rho_{ei}^{SAT} + \rho_{e(i+1)}^{SUN}) && && \text{(6.17c)} \\
& - W^{IW} \cdot \sum_{i \in \mathcal{I}} \sigma_{ei} && && \text{(6.17d)} \\
& - W^{IO} \cdot \sum_{i \in \mathcal{I}} \phi_{ei} && && \text{(6.17e)} \\
& - W^{CD} \cdot \sum_{i \in \mathcal{I}} \pi_{ei} && && \text{(6.17f)} \\
& + W^P \cdot \sum_{t \in \mathcal{T}} P_{et} \cdot \sum_{c \in C} y_{cet} && e \in \mathcal{E} && \text{(6.17g)}
\end{aligned}
$$

$$g \leq f_e \qquad e \in \mathcal{E} \tag{6.18}$$

Constraints on Variables

Constraints (6.19) to (6.26) ensures that the binary variables are kept binary, while constraints (6.27) to (6.29) makes sure that the integer variables are kept non-negative. Finally, constraints (6.30) and (6.31) states that f_e and g are free variables. In some cases the fairness scores may become negative, depending on weighting and/or the scheduling problem at hand. Thus, these fairness variables have to be free.

$$x_{es} \in \{0,1\} \qquad e \in \mathcal{E},\ s \in \mathcal{S} \setminus \mathcal{S}_e^{INV} \tag{6.19}$$

$$y_{cet} \in \{0,1\} \qquad c \in \mathcal{C},\ e \in \mathcal{E},\ t \in \mathcal{T} \tag{6.20}$$

$$w_{er} \in \{0,1\} \qquad e \in \mathcal{E},\ r \in \mathcal{R} \tag{6.21}$$

$$\gamma_{ei} \in \{0,1\} \qquad e \in \mathcal{E},\ i \in \mathcal{I} \tag{6.22}$$

$$\rho_{ei}^{SAT}, \rho_{e(i+1)}^{SUN} \in \{0,1\} \qquad e \in \mathcal{E},\ i \in \mathcal{I}^{SAT} \tag{6.23}$$

$$\sigma_{ei} \in \{0,1\} \qquad e \in \mathcal{E},\ i \in \mathcal{I} \tag{6.24}$$

$$\phi_{ei} \in \{0,1\} \qquad e \in \mathcal{E},\ i \in \mathcal{I} \tag{6.25}$$

$$\pi_{ei} \in \{0,1\} \qquad e \in \mathcal{E},\ i \in \mathcal{I} \tag{6.26}$$

$$\delta_{ct}^{+},\ \delta_{ct}^{-} \geq \mathbb{Z}^{+} \qquad c \in \mathcal{C},\ t \in \mathcal{T} \tag{6.27}$$

$$\lambda_e \geq \mathbb{Z}^{+} \qquad e \in \mathcal{E} \tag{6.28}$$

$$\mu_{ct} \geq \mathbb{Z}^{+} \qquad c \in \mathcal{C},\ s \in \mathcal{S},\ t \in \mathcal{T} \tag{6.29}$$

$$f_e = free \qquad e \in \mathcal{E} \tag{6.30}$$

$$g = free \tag{6.31}$$

6.4 Modeling Choices and Insights

This section aim to provide insight and a brief discussion on the modeling choices made when designing the model presented throughout this chapter. An overview of the modeling choices to be discussed are presented in Table 6.1.

Table 6.1: Prominent modelling choices for the MIP model

Choice 1	Modelling of daily and weekly rest
Choice 2	Not restricting symmetry directly

While required weekly rest is ensured through modelling of explicit off-shifts, the required daily rest is handled implicitly by stating invalid shifts, shift combinations and shift sequences. It is considered more efficient to handle daily off-shifts implicitly, instead of generating all off-shifts enabling daily rest. The latter could result in a very large set of daily off-shifts, heavily influencing the performance of the MIP model. Additionally, in some cases the demand structure naturally ensures that the required daily rest is satisfied, making potential daily off-shifts redundant. For handling of weekly rest, the explicit approach is considered most efficient. The set of potential weekly off-shifts are naturally much smaller than for the potential daily off-shifts, as weekly off-shifts span a considerably larger interval of time. In addition, explicitly generating the weekly off-shifts enables efficient evaluation of the amount of weekly rest allocated to each employee. This is important as we are aiming to maximize the duration of weekly rest. The opposite holds for daily rest, where it is sufficient to say that daily rest is fulfilled.

The final choice to be discussed is the absence of explicit formulations of symmetry breaking restrictions. When dealing with personnel scheduling problems, there is a possibility that employees are symmetric. In such cases, means to reduce symmetry could help to narrow the solution space, and thus enhance the performance of the MIP model. It is however assumed that the MSP entails employee specific information that makes the staff less symmetric. Each employee has a specific contractual agreement, stating the number of contracted hours and blocked hours. In addition, each employee has a specific set of competencies, specific requirements to daily and weekly rest, a specific set of preferences and a certain employee specific day defined. All these elements are likely to make the employees less symmetric, and explicit symmetry breaking restrictions are thus not addressed in the MIP model.

Chapter 7

The Heuristic Approach

This chapter introduces a Parallel Adaptive Large Neighborhood Search (PALNS) as a new solution approach for solving rostering problems. The basis of this implementation is the Adaptive Large Neighborhood Search (ALNS), which is described in detail in Section 7.1. Section 7.2 discuss the parallelization of the ALNS, denoted Parallel Adaptive Large Neighborhood Search (PALNS)

7.1 Adaptive Large Neighborhood Search

This section introduces an ALNS heuristic with a basis in the article written by Pisinger and Ropke (2007), for solving the MSP. First, the outline of the ALNS framework used in this thesis is presented in Section 7.1.1. The choice of feasibility is discussed in Section 7.1.2. Next, a short description of the construction of the initial solution is given in Section 7.1.3, before two relevant acceptance criteria are presented in 7.1.4. Selection of operators is discussed in Section 7.1.5. Finally, the implemented destroy operators, repair operators and local search operators are introduced in Section 7.1.6, 7.1.7 and 7.1.8 respectively.

7.1.1 The ALNS Framework

The search process of the ALNS is to alternate between destroying certain aspects of the current solution, and then repair the destroyed parts in an improving manner. The special component of the ALNS, extending it from an LNS, is the use of multiple destroy and repair operators, together with adaptive weights, to choose the most suitable operator for the problem instance. This thesis expands on this by including additional local search operators.

Algorithm 3 outlines the Adaptive Large Neighborhood Search as implemented in this thesis. An initial solution is constructed through an implemented MIP model, discussed in detail in Section 7.1.3. The constructed solution is guaranteed feasible and is distributed to the three variables, sol_{best}, $sol_{candidate}$ and $sol_{current}$. The variable $s_{candidate}$ is the incumbent solution, while $s_{current}$ is the currently accepted solution. The search is exe-

cuted from $s_{current}$ in each iteration. The variable s_{best} refers to the best feasible solution. The algorithm runs until a stopping criterion is met, which in this thesis is implemented as a runtime restriction. Both a destroy and repair operator is chosen based on the previous success of the operators. The selection is discussed in Section 7.1.5, while the destroy and repair operators are discussed in Section 7.1.6 and 7.1.7 respectively. Applying the destroy and repair operators to the current solution produces a new candidate solution. This thesis implements a local search step, not usually included in the ALNS framework. The selection of the local search is presented in Section 7.1.5, while the implemented operators are presented in Section 7.1.8. The candidate solution is then accepted or rejected in regards to one of the acceptance criteria described in Section 7.1.4. Finally, the weights corresponding to the applied destroy operator d and repair operator r are updated according to their performance.

Algorithm 3 Adaptive Large Neighborhood Search

Input: problem instance I
Output: Schedule for problem instance I

```
1:  sol_0 = constructSolution()                                  ▷ Section 7.1.3
2:  sol_best = s_0
3:  sol_candidate = s_0
4:  sol_current = s_0
5:  while stopping criterion not met do
6:      d = selectDestroyOperator()                              ▷ Section 7.1.5
7:      r = selectCoupledRepairOperator(d)                       ▷ Section 7.1.5
8:      sol_candidate = r(d(s_current))
9:      if LocalSearchAccepts(sol_candidate) then                ▷ Section 7.1.5
10:         RunLocalSearch(sol_candidate)
11:     if accepts(sol_candidate, sol_current, sol_best) then    ▷ Section 7.1.4
12:         sol_current = sol_candidate
13:     if F(sol_candidate) > F(sol_best) then
14:         sol_best = sol_candidate
15:     updateOperatorWeights()
16: return sol_best
```

7.1.2 Feasibility and Objective Function

The ALNS is allowed to accept new infeasible solutions if they look promising relative to the current solution, as defined by the acceptance criteria. The reasoning behind this decision is that the optimal solution to linear mixed integer-programming problems is located at the intersection between feasibility and infeasibility. By allowing infeasible solutions, the algorithm can search on both sides of this intersection and hopefully navigate with greater ease between high-quality feasible solutions. The search space for our problem is quite large, and restricting the ALNS only to use feasible solutions could cause too many constraints on the problem, making the search more difficult.

Although the ALNS allows infeasible solutions, the architecture of the implementation prevents the violation of some constraints. When assigning a shift to an employee, the algorithm simultaneously assigns associated demand to cover. This makes it impossible to cover multiple competencies, while also mapping shift to demand, reflected by constraints (6.7) and (6.6), respectively. Additional affected constraints are maximum one shift per day, and that an employee should not cover demand while assigned to a weekly off-shift, presented in the constraints (6.5) and (6.11), respectively.

To prevent the accepted infeasible solutions to deviate far from the boundary of feasible solutions, a penalty expression $p(x)$ is added to the heuristic objective function, with a weight parameter presented in Chapter 8. The general problem is therefore transformed from the model presented in Chapter 6, to:

$$\begin{aligned} \text{Maximize} \quad & F(x) = f(x) - W^{IP} \cdot p(x) \\ & x \in \mathcal{X} \end{aligned} \tag{7.1}$$

where $p(x)$ is the sum of violations of hard constraints, and W^{IP} is the penalty weight. The penalty weight parameter decides the impact of infeasibility on the objective value. Consequentially, one can increase the level of infeasible solutions by reducing the parameter value before running the heuristic.

The objective function $f(sol)$ is equal the one presented in Chapter 6. For the convenience of the reader the objective function is presented in Equation (7.2). Here the variable x is substituted by $sol_{candidate}$, which denotes the incumbent solution used in the heuristic. To reduce computation time, only the affected variables of the repair and destroy operators are included in the computation of the objective value.

$$\begin{aligned} f(sol_{candidate}) = \quad & \sum_{e \in \mathcal{E}} f_e \\ & + W^F \cdot A^F \cdot g \\ & - A^D \cdot \sum_{c \in \mathcal{C}} \sum_{t \in \mathcal{T}} (W^{D+} \cdot \delta^+_{ct} \; + \; W^{D-} \cdot \delta^-_{ct}) \end{aligned} \tag{7.2}$$

7.1.3 Construction

The construction of an initial solution for the ALNS is achieved through a simplified version of the MIP-model presented in Chapter 6, using the shifts generated in the shift design stage presented in Chapter 5. MIPs are, generally speaking, more costly to run than heuristic methods. Nevertheless, the MIP construction model has shown to be highly effective, even on large problem sizes. This made the implementation of the ALNS easier as it allowed for reuse of existing code, and the MIP guarantees that the constructed solution is feasible, and therefore a promising candidate solution.

To make the MIP construct the initial solution as fast as possible while simultaneously making sure the solution provides a good basis for the neighborhood search, several measures have been taken. First, the soft constraints from the MIP model are omitted, as the constructed solution is only required to be feasible. This affects constraints (6.13) - (6.18) as well as preferences and blocked hours. A full model is presented in Appendix A.4. Secondly, the objective function is modified to simplify the search. The modified objective can be seen in Equation (7.3). From this objective function, the MIP allocates just enough shifts to cover the minimum demand faced, constructing a feasible solution.

$$\text{Minimize} \quad f'(x) = \sum_{c \in \mathcal{C}} \sum_{e \in \mathcal{E}} \sum_{t \in \mathcal{T}} y_{cet} \tag{7.3}$$

7.1.4 Acceptance Criteria

The acceptance criteria is a high-level method to direct the neighborhood search. If an incumbent solution $s_{candidate}$ is accepted by the criteria, the current solution is discarded in favor of the incumbent solution. The next iteration of the ALNS will then use $s_{candidate}$ as both the basis for the destroy-repair operation and the following acceptance test.

If the candidate solution improves upon the best solution, and the solution is feasible, the candidate solution becomes the new best solution. This is a design decision made to balance two considerations:

(i) As mentioned, many of the best solutions are located at the intersection between feasibility and infeasibility. Thus, it makes sense to allow the algorithm to search on both sides of this intersection, and the candidate solution should be able to take on infeasible values.

(ii) Only feasible solutions have a practical value for the users of the algorithm. The best solutions should be restricted to feasible solutions.

There exists a large number of algorithms suited for the task as the acceptance criterion. Hill climbing was used in the original implementation of LNS by Shaw (Pisinger and Ropke, 2019). Meanwhile, Ropke and Pisinger used Simulated Annealing in the first implementation of ALNS, and it has been the most common choice within the framework (Santini et al., 2018). Santini et al. (2018) provides an extensive comparison of different criteria based on the implementation of PALNS from (Ropke and Santini, 2019). According to their research variants of Simulated Annealing, Record to Record Travel and Threshold Acceptance consistently outperform the rest of the criteria on a set of four different problems. Furthermore, Random Walk and Hill Climbing consistently yield the worst results. The difference between the worst and best acceptance criteria has been shown to be up to 2.5 percentage points for the obtained gaps. We have chosen to implement Hill Climbing and Record to Record Travel.

Hill Climbing

Hill Climbing is the simplest acceptance criterion and follows a pure greedy strategy by only accepting improving solutions. The primary benefit of this method is the ease of

Algorithm 4 Hill Climbing

Input: Solution $sol_{candidate}$, Solution $sol_{current}$
Output: isAccepted
1: **if** $F(sol_{candidate}) > F(sol_{current})$ **then**
2: $\quad isAccepted = True$
3: **else**
4: $\quad isAccepted = False$
5: **return** $isAccepted$

use, as the algorithm is trivial to implement and does not introduce any working in tuning parameters. The drawback is that the criterion easily can get stuck in local maxima.

Record to Record Travel

Algorithm 5 Record to Record Travel

Input: Solution $sol_{candidate}$, Solution sol_{best}
Output: $isAccepted$
1: $threshold = getCurrentThreshold()$
2: $diff = F(sol_{best}) - F(sol_{candidate})$
3: **if** $diff < 0$ **then**
4: $\quad isAccepted = True$
5: **else**
6: $\quad isAccepted = |diff|/|F(sol_{candidate})| \leq threshold$
7: $updatedCurrentThreshold()$
8: **return** $isAccepted$

Like Hill Climbing, Record to Record Travel will always accept an improving solution. However, it can also accept worse solutions. Record to Record Travel accepts solutions within an absolute distance from the best solution. When Record to Record Travel reaches a threshold of 0 it behaves identically to Hill Climbing. The threshold is updated linearly, that is, a constant step length is withdrawn from the current threshold in each iteration. As a consequence, Record to Record Travel will gradually be more restrictive in how much worse a solution can be and still be accepted. Note that when the current threshold reaches 0, Record to Record Travel will behave like Hill Climbing. The benefit of Record to Record Travel is its ability to facilitate a diverse search in the early stages, before transitioning towards an intensified search. This should make it less prone to stagnation than Hill Climbing. The drawback is that it can be challenging to find suitable values for the input parameters. The criterion takes start threshold, end threshold and step length as input. Higher threshold results in a more diverse search.

7.1.5 Selection of Operators

Selection of Repair And Destroy Operators

The ALNS is implemented with coupled neighborhoods, meaning that each destroy operator is coupled with a subset of the repair operators. This coupling enables the use of more custom-tailored repair heuristics, thus exploiting more of the structure of the subproblem that a given destroy operator creates.

Having k destroy operators we can define the set of destroy operators $D = \{d_i | i = 1, 2...k\}$. The weights of the destroy operators are denoted $w(d_i)$. For each destroy operator d_i a coupled set of l repair operators is defined $R^i = \{r^i_j | j = 1, 2...l\}$. The weights of the coupled repair operators are denoted $w(r^i_j)$. Note that one given repair operator will be assigned individual weighting variables for each associated destroy operator. Initially, all weights are equal, meaning that the probability distribution for selection of operators is uniform. The probabilities for selecting a destroy operator, then selecting a coupled repair operator is given in the Equations (7.4) and (7.5) respectively.

$$p(d_i) = \frac{w(d_i)}{\sum_{n=1}^{l} w(d_n)} \qquad (7.4) \qquad\qquad p(r^i_j) = \frac{w(r^i_j)}{\sum_{n=1}^{k} w(r^i_n)} \qquad (7.5)$$

To select a destroy operator, a random draw is performed by using the Roulette Wheel principle. Next, a new draw is performed to select a repair operator among the operators coupled with the selected destroy operator.

Operator Weight Update

After the destroy and repair operations are completed, the respective operator weights are updated. First, the appropriate ψ-parameter is selected. The weights are then updated according to Equation (7.6), based on the reward and the decay parameter η. The decay parameter controls the weighting of historical performance versus the most recent result. The parameters are presented below.

Reward parameters

η : The decay for operator performance.

ψ_{best} : Reward for finding a candidate solution that improves on the best solution found. The candidate solution must be feasible.

ψ_{better} : Reward for finding a candidate solution that improves on the current solution, regardless of feasibility.

$\psi_{accepted}$: Reward for finding a candidate solution that is accepted, regardless of feasibility.

$\psi_{rejected}$: Penalty for finding a candidate solution that is rejected, regardless of feasibility.

$$w_i = \eta * \psi + (1 - \eta) * w_i \tag{7.6}$$

This behavior, often denoted adaptiveness, aims to increase the likelihood of selecting operators that have performed well with regards to finding improved solutions, and decrease the likelihood of selecting operators with poor results. Because of this bias, an ALNS can afford to include many operators, even though some of which may only be useful for certain situations (Pisinger and Ropke, 2019). We have implemented a large number of varied operators to take advantage of this characteristic of ALNS.

Selection of Local Search Operators

To reduce computation time used on local search operators, the operators are only run when certain criteria are fulfilled, depending on the local search operator. The criteria guarantees that the operators are only run when there is a potential for improvement. An important aspect of this idea is that all local search operators implemented, targets a specific part of the solution that is infeasible. Thus the illegal week swap is only run when the solution violates the weekly rest constraint. The distribute illegal contracted hours operator runs when at least one employee violates the contracted hours constraint. Additionally, attempting to improve a solution that largely deviates from the current solution in a negative direction, most likely leads to a wasted run. All local search operators are therefore run only when the gap between the current solution and candidate solution is less than 100%.

7.1.6 Destroy Operators

In this section, we introduce the different destroy operators used to remove certain aspects of a solution, which would later be rebuilt. Common strategies for destroy operators include critical destroy and random destroy. A critical destroy operator will aim to delete the worst part of the solution, while a random destroy will use randomness to delete parts of the solution (Pisinger and Ropke, 2019). Both strategies have been implemented in this thesis. In this thesis, two coupled neighborhoods have been implemented based on week removal and employee removal. These couplings define the contents of the destroy specific set, required in all repair operators.

According to Pisinger and Ropke (2019), ”the most important choice when implementing the destroy method is the degree of destruction. A large destroy size may lead to repeated re-optimization and few iterations. In contrast, a small destroy size might restrict the diversity of the search, reducing the advantages of the large neighborhood”. This thesis implements a static destroy size associated with each destroy operator. For all week destroy operators, the destroy size constitutes the destruction of one week. For the employee destroy operators, the destroy size ranges from two to four employees, depending

on the operator. Before describing the implemented operators, variables used in the worst week scoring function, presented in Equation (7.7) have to be defined. The variables are presented below.

Variables of week destroy functions

v_j^{IO} : Total number of violations of isolated off days in week j

v_j^{IW} : Total number of violations of isolated working days in week j

v_j^{CD} : Total number of violations of consecutive days in week j

v_j^{PW} : Total number of violations of partial weekends in week j

q_j : Total number of preferences granted in week j

w_{ej} : Maximum weekly rest in week j for employee e

θ_j : Violations of hard constraints in week j

Algorithm 6 illustrate the procedure of the week destroy operator, while Algorithm 7 outlines the employee destroy operators. The candidate solution is passed as input denoted sol, together with the destroy size k. In the case of week destroy operator, k denotes the number of weeks, while in employee destroy operator k denotes the number of employees to be destroyed. The output of the operators is a destroyed solution sol', in addition to the destroy specific set d containing the specific weeks or employees destroyed. Destroying a week imply that all shift assignments for that week or the employee are removed. A modeling choice has been made in implementing destroy operators with small changes as individual operators. This is done with the adaptiveness of the ALNS in mind, allowing the ALNS to select the most successful operator for each problem instance.

Worst Week Destroy Operator

The worst week destroy operator aims at destroying the worst week of the solution in regards to the scoring function defined by Equation (7.7). The values of the included weights are displayed in Chapter 8.

Algorithm 6 Worst Week Destroy Operator

Input: Solution sol, destroy size k
Output: Destroyed solution sol', destroy specific set d

1: $selectedWeeks = getWorstWeeks(s, k)$
2: **for** j in $selectedWeeks$ **do**
3: $destroyedShifts$ = all shifts in j
4: sol' = sol - $destroyedShifts$
5: **return** sol', d

$$\arg\min_{j \in \mathcal{J}} \left\{ \begin{array}{l} \sum_{e \in \mathcal{E}} w_{ej} - \sum_{c \in \mathcal{C}} \sum_{t \in \mathcal{T}_j} (D_{ct} - \sum_{e \in \mathcal{E}} y_{cet}) - W^{DID} \cdot \lambda_j^- - W^{HC} \cdot \theta_j - \\ W^{EIO} \cdot v_j^{IO} - W^{EIW} \cdot v_j^{IW} - W^{EPW} \cdot v_j^{PW} - W^{ECD} \cdot v_j^{CD} \end{array} \right\} \tag{7.7}$$

Random Weighted Week Destroy Operator

This operator utilizes a weighted random choice to select the number of weeks determined by the destroy size k. The weights are based on the week score calculated using Equation (7.7).

Random Week Destroy Operator

The operator destroys the number of weeks determined by the destroy size k randomly.

Random Weekend Destroy Operator

Having a high number of partial weekends could potentially lead to highly reduced objective values and unsatisfactory schedules for the employees. As violations of partial weekends are easy to pinpoint, and in most cases manageable, we have decided to implement an operator targeting these violations. This operator removes all employees for the number of weekends determined by the destroy size k. The selection of the weekends is done randomly.

Worst Employee Destroy Operator

The general pseudocode for the employee destroy operators is shown in Algorithm 7. The employee destroy operators are coupled with a different set of repair operators than the week operators. This necessitates outputting a destroy specific set, containing the destroyed employees, instead of destroyed weeks. The operator greedily selects the k worst employees and the k best employees, based on their fairness score, which is presented in constraints (6.17). It then removes all shifts belonging the selected employees over the entire planning period, from the incumbent solution $sol_{candidate}$. The destroyed employee is then added to the destroy specific set. The reason for destroying both worst and best is in an attempt to distribute the unpopular shifts more evenly, resulting in higher lowest fairness values.

Worst Contracted Destroy Operator

This operator works exactly like the worst employee destroy operator, except for the selection of the employees. Whereas the worst employee destroy operator takes the entire fairness score into account when selecting employees, this operator relies only on the deviation from contracted hours for each employee over the entire planning period.

Algorithm 7 Worst Employee Destroy Operator

Input: Solution sol, destroy size k
Output: Destroyed solution sol', destroy specific set d
1: $selectedEmployees = getEmployees(s, k)$
2: **for** e in $selectedEmployees$ **do**
3: $d.insert(e)$
4: **for** $shiftAllocation$ in $getShiftAllocation(employee)$ **do**
5: $destroyedShifts$ = all shift allocations for employee e
6: sol' = sol - $destroyedShifts$
7: **return** sol', d

Random Weighted Employee Destroy Operator

This operator resembles the random weighted week destroy operator. It utilizes the fairness score, calculated for the entire planning period, to associate a probability to each employee before a sample of k employees are drawn at random.

Random Employee Destroy Operator

Similarly to the worst employee destroy operator, the random employee operator selects k employees and removes all their shift assignments. The selected employees are picked randomly.

7.1.7 Repair Operators

This section presents the repair operators used in the ALNS. A total of 10 repair operators are implemented, which explores different neighborhoods. Algorithm 8 presents the architecture of which all operators, except the pure MIP-operator, are built on. As all operators developed are implemented as coupled neighborhood operators, a destroy specific set is needed in the repair operator. $getDemandCompetency$ is a function that decides which competency to assign the employees. This function is used in every operator, except for the pure MIP operator. Between the hybrid operators and heuristic operators, the difference is caused by the combinations of the two functions $getShiftsSet()$ and $getEmployee()$, presented on lines 2 and 6, respectively. The functions used will be presented with respect to each operator.

$$\arg\max_{c \in C} \left\{ D_{ct} - \sum_{e \in \mathcal{E}} y_{cet} \right\} \tag{7.8}$$

The operators are presented depending on their coupled neighborhood, starting with week repair operators, as most operators belong to this class before the single employee repair operator is introduced. A high number of week repair operators is implemented as preliminary tests showed a higher repair potential when compared to the repair operators coupled with employee removal operators. In addition to a hybrid operator and multiple heuristic

Algorithm 8 Outline Repair Operator

Input: Destroyed solution sol, Destroy specific set d
Output: Repaired schedule sol'

```
while stopping criterion not met do
    s = getShift()                                   ▷ Equation (7.9)
    for t covered by s do
        Y_t.insert(getDemandCompetency())   ▷ Equation (7.8), SR extension 5.4
    E^POS = set of employees not working the day of shift s
    e = getEmployee()                                ▷ Equation (7.10), (7.11)
    in sol' assign e to s covering demands Y
return sol'
```

operators, a MIP repair operator is developed. It is coupled with the week removal operators. Within the couplings, the operators are presented in an order starting with the simplest to the most complex operator.

To discuss the more complex scoring functions, we first have to define variables that hold the violations of both hard and soft constraints used in the calculations, as well as the current achieved weekly rest and preferences. The variables are defined below.

Variables of week repair functions

v_{es}^{DR} : Violations of daily rest for employee e being assigned shift s

v_{es}^{WR} : Violations of weekly rest for employee e being assigned shift s

v_{es}^{IO} : Violations of isolated off days for employee e being assigned shift s

v_{es}^{IW} : Violations of isolated working days for employee e being assigned shift s

v_{es}^{CD} : Violations of consecutive days for employee e being assigned shift s

v_{es}^{PW} : Violations of partial weekends for employee e being assigned shift s

q_{es} : Additional preferences granted for employee e being assigned shift s

w_{es} : Maximum weekly rest achievable for employee e being assigned shift s

Week focus repair operators

The week repair operators focus on repairing the shift assignment of the destroyed weeks. All employees available for the problem are considered when assigning shifts. The destroy specific set d contains the week or weeks destroyed.

Week Greedy Contracted Hours Repair Operator

The Week Greedy Contracted Hours Repair Operator uses Equation (7.9) to choose the shift that has the highest deviation from ideal demand, in the week(s) specified in the

destroy specific set d. It then uses Equation (7.10) to greedily select the employee with the highest deviation from their contracted hours to cover the shift.

$$\arg\max_{s \in S} \left\{ \left(\sum_{c \in \mathcal{C}} \sum_{t \in s} (D_{c,t} - \sum_{e \in \mathcal{E}} y_{c,e,t}) \right) - v \right\} \tag{7.9}$$

$$\arg\max_{e \in \mathcal{E}_s^{POS}} \left\{ B_e - \sum_{s \in S} (len(s) \cdot x_{es}) \right\} \tag{7.10}$$

Week Greedy Objective Repair Operator

An operator that simulates the score of assigning a shift to an employee. It selects shifts using Equation (7.9), as the previous operator but utilizes a more complex scoring method to select the employees greedily. The scoring function is presented in Equation (7.11), which is meant to resemble the objective function of the heuristic.

$$\arg\max_{e \in \mathcal{E}_s^{POS}} \left\{ \begin{array}{l} W^{VDR} \cdot v_{es}^{DR} + W^{VPW} \cdot v_{es}^{PW} + W^{VIO} \cdot v_{es}^{IO} + \\ W^{VIW} \cdot v_{es}^{IW} + W^{VWR} \cdot \mathrm{w}_{es} + W^{VCD} \cdot v_{es}^{CD} + W^{P} \cdot q_{es} \end{array} \right\} \tag{7.11}$$

Week Greedy Demand At Shift Based Operator

One of the most challenging problems faced by the operators is to select combinations of shifts to assign, that are able to satisfy the ideal demand faced for every time period. The shift selection is in the previous operators carried out by a greedy choice. This has proven efficient on most problem instances. However, some demand structures require a different approach. This operator takes advantage of the demand for shifts calculated in the SR extension presented in Chapter 5. The solution to this MIP model is a set of shifts with a corresponding number of employees needed to cover the demand faced for each time period throughout the planning horizon. The operator greedily selects the shifts that have the highest demand for employees from the MIP solution. When selecting an employee to be assigned the shift, it does a greedy choice using the scoring function presented in Equation (7.11).

Week Random Demand At shift Based Operator

Similar to the previously discussed operator, this operator also uses the demand for shifts-set created by the SR extension. It differs from the previous operator, in that the selection of shifts is made using a weighted random choice instead of being purely greedy. The weights are given by the demand for each shift given by the SR extension.

Week demand per shift heap repair

In addition to constructing a candidate solution, SDH also proposes how many employees to allocate to each shift to minimize deviation from ideal demand. In effect, this is a conversion from demand per time unit, to demand per shift. This operator selects the

remaining shift with the highest demand. The employee with the most remaining working hours is allocated to this shift until the entire associated demand is covered. The next shift is then selected until all the demand is covered.

Week demand heap repair

This operator selects shifts according to Equation (7.12). After a shift is found, sufficiently, many employees are allocated to cover the smallest minimum demand over the relevant time period. Equation (7.13) shows the calculation. The employees with the most unused contracted hours are used. Note that $\mathcal{T}_s^S$ represents the set of time periods covered by shift s, while V_s^S represents the duration of the shift.

$$\sum_{t \in \mathcal{T}_s^S} \sum_{c \in C} D_{ct} - V_s^S \tag{7.12}$$

$$\arg\min_{t \in \mathcal{T}_s^S} \sum_{c \in C} D_{ct} \tag{7.13}$$

Greedy Fairness Score Based Operator

An operator created to directly target an even distribution of unpopular shifts among the employees, using the fairness score. As the fairness score is computationally heavy to calculate, it is impractical to use it in all operators. The decision was then to incorporate the functionality into a separate operator to reduce computation time. The shifts are selected in the same fashion as for the demand for shift operators, described above. The employees are chosen greedily based on current fairness value for all employees, in addition to the increase in fairness value by assigning the selected shift to an employee. Equation (7.14) presents the scoring function used in the greedy choice. The last two expressions are the penalties of breaking the daily rest and weekly rest constraints, respectively. Notice that the weights used in the scoring function are the weights presented in Chapter 8 and used in the objective function of the mathematical model of Chapter 6.

$$\arg\max_{e \in \mathcal{E}_s^{POS}} \left\{ \begin{pmatrix} W_r^R \cdot w_{er} + W^{PW} \cdot v_{es}^{PW} + W^{IO} \cdot v_{es}^{IO} + \\ W^{IW} \cdot v_{es}^{IW} + W^{CD} \cdot v_{es}^{CD} + W^P \cdot q_{es} \end{pmatrix} - v_{es}^{DR} - v_{es}^{WR} \right\} \tag{7.14}$$

Hybrid Operator

Another method of selecting shifts to be assigned to employees is by utilizing an implemented MIP model. The idea is similar to the SR extension introduced in Chapter 5 in that it finds the optimal number of employees to assign to each shift, in order to cover the faced demand. The model utilizes the destroy specific set containing the week or weeks destroyed to constrict the complexity of the implemented model. The reader is referred to Appendix A.5.2 for a complete definition of the MIP model used in this operator. To assure decent computation time per iteration, the implemented MIP model has a runtime restriction of 5 seconds. Having selected the set of shifts to assign, the employees allocated using the greedy scoring function from Equation (7.11).

Pure MIP operator

The pure MIP operator is the only operator that is not built according to the pseudocode presented in Algorithm 8. Instead, the operator is implemented with the Gurobi Optimizer and is a slightly modified version of the mathematical formulation introduced in Chapter (6). The constraints (6.2) - (6.9) and (6.13) - (6.16) are implemented directly. In addition, constraints (6.12) is modified to become a soft constraint in both directions, allowing the use of more, and less than the total contracted hours available. To allow using more than the number of contracted hours for an employee, λ_e^+ is defined, which holds the number of hours over the contracted hours used for employee e. Notice that constraints (6.10) and (6.11) are not included. These constraints have been substituted for a constraint restricting the total number of consecutive working days for an employee to N days. Throughout this thesis, N is fixed at six, as we use static destroy sizes. This is done as every employee would need at least one day without work to satisfy the weekly rest constraint. The modified objective function is presented in Equation (7.15). The weight W^{OCH} is set to be 1.2. For a complete definition of the mathematical model used, the reader is referred to Appendix A.5.1. To ensure moderately fast iterations, the implemented MIP model is restricted to run for 5 seconds.

Pisinger and Ropke (2019) discusses both advantages and disadvantages of implementing an optimal repair operator. The disadvantages mainly focus on the difficulties of leaving local optima. The advantage, however, is that such operators may lead to high-quality solutions in a few iterations. This thesis implements a high number of operators able to diversify the search, which should be able to mitigate some of the disadvantages expressed by Pisinger and Ropke (2019).

$$\begin{aligned}\max z = \quad & -W^{OCH} \cdot \sum_{e \in \mathcal{E}} \lambda_e^+ - W^{PW} \cdot \sum_{i \in \mathcal{I}^{SAT}} (\rho_{ei}^{SAT} + \rho_{e(i+1)}^{SUN}) \\ & - W^{IW} \cdot \sum_{i \in \mathcal{I}} \sigma_{ei} - W^{IO} \cdot \sum_{i \in \mathcal{I}} \phi_{ei} \\ & - W^{CD} \cdot \sum_{i \in \mathcal{I}} \pi_{ei} + W^{P} \cdot \sum_{t \in \mathcal{T}} P_{et} \cdot \sum_{c \in C} y_{cet} \\ & - \sum_{c \in \mathcal{C}} \sum_{t \in \mathcal{T}} (W^{D+} \cdot \delta_{ct}^+ \; + \; W^{D-} \cdot \delta_{ct}^-)\end{aligned} \tag{7.15}$$

Employee Repair Operator

What characterizes the employee repair operator is that the destroy specific set contains the individual employees who have been removed from the solution. The idea behind the operator is to focus on specific employees mainly to increase their fairness values.

Employee Greedy Objective Repair Operator

The employee greedy operator shares similarities to the week greedy objective repair operator presented above. The shifts are selected using the greedy shift scoring function in

equation (7.9). Additionally, the employee score used is the one presented in equation (7.11). The only difference between the two operators is the selected set of shifts to assign to employees. While the week focus operators are only allowed to assign shifts belonging to the destroyed week, the employee focus operator is allowed to assign shifts belonging to the entire planning period.

7.1.8 Local Search Operators

Local Search Operators are a novel supplement to the regular destroy and repair operators of ALNS and can be utilized to enhance the search. Pisinger and Ropke (2019) considers the use of ordinary local search heuristics to compete with the other destroy and repair neighborhoods. They argue that such operators could ensure that a thorough investigation of the solution space, close to the current solution, is made from time to time. The operators implemented in this thesis are based on this idea, although with a different approach than presented by Pisinger and Ropke (2019). Our implemented operators aim to make relatively simple modifications to promising, but infeasible solutions, thus improve the quality of the incumbent solution. This follows the assumption that optimal solutions exist on the boundary between feasible and infeasible solutions. Preliminary tests of the implemented repair operators revealed frequent violations of the weekly rest and contracted hours restrictions. Thus, the local search operators implemented to target these problem areas. It is important to notice that the local search operators do not guarantee a feasible solution. The reason is to shorten the computation time and ensure the operator does not get stuck when no improvement is possible.

Illegal week swap

Algorithm 9 outlines a local search operator targeting violations of weekly rest. The illegal week swap operator attempts to assign continuous weekly rest to an employee in order to satisfy the weekly rest constraint. The idea is to remove one shift from the employee violating the constraint by assigning it to another employee, effectively swapping an off-shift with a work shift. It loops through all employee/week combinations that violate the weekly rest constraint exactly once. It then tries to remove every shift assigned to that employee in that particular week, and greedily select the best combination of a new employee/shift combination to swap. The $employeesFixed$-set ensures that an employee/shift combination that has been repaired in this iteration is not destroyed later. As the operator only loops through all violations once and does not ensure that the swap is feasible, the operator could potentially lead to new infeasible solutions.

Distribute illegal contracted hours

The repair operators decide on shifts to assign, separately from the employee it should be assigned to. This could lead to assigning more than the allowed contracted hours. To mitigate this effect, a local search operator has been implemented, that distribute the excess contracted hours from an employee to the rest of the employees with additional working hours available. An outline of the process is displayed in Algorithm 10. The algorithm loops over all employees, violating the contracted hours constraint. It then goes through

Algorithm 9 Illegal Week Swap

Input: solution sol
Output: new solution sol'
1: Initialize empty set $employeesFixed$
2: **for** employee e and week j in s violating weekly off shift **do**
3: $availableShifts =$ all shifts assigned to employee e in week j
4: Initialize empty $shiftScore$ list
5: **for** s in $availableShifts$ **do**
6: $\mathcal{E}_s^{POS} =$employees not working on day of shift s and not in $employeesFixed$
7: $shiftScore.insert(getEmployee())$ ▷ Equation (7.11)
8: $selectedEmployee, s = \arg\max_{e \in \mathcal{E}_s^{POS}} shiftScore[e]$
9: sol' = Assign s to employee $selectedEmployee$ in sol'
10: $employeesFixed.insert(e, s)$
return s'

the employees' assigned shifts and attempts to swap the shift with another employee working a shorter shift, on the same day. If it finds a shift that perfectly matches the excess use of contracted hours, this shift is swapped, and we continue to the next employee violating the constraint. Otherwise, a random choice is made on which employee the algorithm should swap the shift between. The algorithm continuous to there are no more shifts to swap from the violating employee before continuing on the next violated employee.

Algorithm 10 Distribute Illegal Contracted Hours

Input: solution sol
Output: new solution sol'
1: **for** employee e in s violating weekly off shift **do**
2: $\lambda^+ =$Number of hours over contracted hours
3: **for** Shift s assigned to e **do**
4: $swapShifts$ = shifts assigned at the same day with duration shorter than s
5: **for** $selectedShift$ in $SwapShifts$ **do**
6: **if** $\lambda^+ - duration(selectedShift) == 0$ **then**
7: Swap employee assignment of s and $selectedShift$
8: **Next** employee
9: $RandomChoice(SwapShifts.pop())$
10: Swap employee assignment of s and $selectedShift$
11: $\lambda^+ = \lambda^+ - duration(selectedShift)$
12: **if** no more shifts assigned to e **then**
13: **Next** employee
14: **return** sol'

7.2 Parallel Adaptive Large Neighborhood Search

Parallel processing can yield significant benefits with regards to processing powers, but one has to take some special considerations to make the most out of this opportunity. It can be especially challenging to utilize all this additional computational power efficiently.

7.2.1 The Parallel Algorithm

The PALNS consists of two main components: a master process responsible for initialization, and subprocesses which perform the actual search. The master process is shown in Algorithm 11. The code initializes a user-defined number of subprocesses with the shared queue, the candidate solution and a unique seed. A seed is a number used to initialize a random generator. Random generators with the same seed will perform identically, and, conversely, random generators with different seed will yield differing results. In practice, this means that each subprocess will get different results from random choice as they all are seeded with a unique number.

Each subprocess performs a slightly modified ALNS-search to take proper advantage of the parallelization. We have named these subprocesses *PALNSWorker*. The pseudo-code is given in Algorithm 12. Note the sharing of solutions on lines 5-11. The subprocess pushes its current solution onto the queue, after popping the first element from it. This avoids a situation where a subprocess shares its current solution with itself. The shared solution is then run through the acceptance criteria, and considered as the new best, in the same way as in conventional ALNS.

Algorithm 11 Parallel Adaptive Large Neighborhood Search

Input: problem instance I, int NumberOfParallels

1: $sol_0 = constructSolution()$
2: $sharedQueue = Queue()$
3: $workerList = list()$
4: **for** n in NumberOfParallels **do**
5: $\quad worker = PALNSWorker(sol_0, sharedQueue, seed = n)$
6: $\quad$ Append $worker$ to $workerList$
7: $\quad$ Start $worker$ as a separate subprocess
8: Wait til every $worker$ finishes
9: $sol_{best} = getBestResult(workerList)$
10: **return** sol_{best}

Algorithm 12 PALNSWorker

Input: Solution sol_0, Queue $sharedQueue$

$sol_{best} = sol_0$
$sol_{candidate} = sol_0$
$sol_{current} = sol_0$
while stopping criterion not met **do**
    if sharing criterion met **then**
        $sol_{shared} = pop(sharedQueue)$
        $push(sharedQueue, sol_{best})$
        if accepts($sol_{shared}, sol_{current}, sol_{best}$) **then**
            $sol_{current} = sol_{shared}$
        if $F(sol_{candidate}) > F(sol_{best})$ **then**
            $sol_{best} = sol_{shared}$
    $d = selectDestroyOperator()$
    $r = selectCoupledRepairOperator(d)$
    $sol_{candidate} = r(d(sol_{current}))$
    if accepts($sol_{candidate}, sol_{current}, sol_{best}$) **then**
        $sol_{current} = sol_{candidate}$
    if $F(sol_{candidate}) > F(sol_{best})$ **then**
        $sol_{best} = sol_{candidate}$
    $updateOperatorWeights()$
return sol_{best}

7.2.2 Important Aspects of Parallelization

The usage of a queue as a sharing mechanism enables efficient and robust sharing of state. However, there are some caveats. One issue is that there is no guarantee that a subprocess does not get stuck with popping solutions that it pushed to the queue in a previous iteration. In practice, this has not been an issue in our preliminary tests, but should it become a problem, one solution might be to have a pair of queues $queueA$ and $queueB$. By alternating whether a subprocess should push to $queueA$ and pop $queueB$, or the other way around this potential problem should be eliminated. Another important consideration is the propagation of solutions. As we are using a queue to share solutions, there might be a considerable delay from a subprocess finds and shares a considerably better solution until it is propagated out to the majority of the other subprocesses. This delay might be substantial if the majority of the subprocesses are stuck in inferior solutions. However, this might also help to diversify the search by allowing the subprocesses to work from different areas of the solution space.

Although parallelization can significantly increase the number of iterations performed, it does not imply that the objective value after a given number of aggregated parallel iterations of PALNS will equal the objective value of a conventional ALNS with the same number of iterations. That is, 10 $PALNSWorkers$ with 1 000 iterations each might reach a different objective value than an ALNS with 10 000 iterations. The main ben-

efit of parallelization is the ability to perform a considerably more diverse search. This might be better suited to some problems, while the benefits might be less relevant for other problems. Having a good sharing strategy is paramount to capitalize on the increase in iterations.

An additional benefit of parallelization is that it enables the heuristic to take many different paths from the same current solution. As each subprocess has different random seeds, they will take different random choices, and thereby exploring the search space differently. They will get different shared solutions from the queue, and thus be stimulated differently. Combined with large destroy sizes, this can reduce the problem of getting stuck in local maxima.

Chapter 8

Test Instances and Parameter Values

In this chapter, the test instances used to evaluate the performance of the implemented solution methods are presented in detail. In addition, the parameter and weighting values used are presented and elaborated on. Section 8.1 presents nine real-life problems provided by Medvind, with an elaboration on the characteristics for each problem. In Section 8.2 the general parameters used when solving the problems are presented, while Section 8.3 elaborates on the weighting parameters used in the SDH, the MIP model and the PALNS.

8.1 Test Instances

In this section, the test instances forming the basis for the analysis performed in Chapter 9 are presented. An overview of the test instances, and their most prominent characteristics, are provided in Table 8.1. Here, the length of the planning horizon, size of staff and the average number of hours with demand per day is specified for each problem, along with the time step of the problems.

The nine test instances are all provided by Medvind and represent real-life problems from a variety of different industries. The problems thus serve as a good basis for testing the generality of the developed solution methods. Each problem is named based on the problem number, length of the planning horizon and the number of employees. Problem P1-8w-9e thus defines problem one, consisting of eight planning weeks and nine employees.

Concerning the problem complexities, it is hard to draw any conclusions solely based on the characteristics presented in Table 8.1. In reality, many aspects are influencing the complexity of a problem. Examples are how often and exactly how the demand changes or how rigid the problem is in terms of requirements to rest or allowed demand deviation. A full analysis of the complexity of the nine real-life problem instances is considered too comprehensive to be included in this thesis. However, the general demand structures for each problem are included in Appendix B. Each problem is represented by a figure, visu-

Table 8.1: Overview of test instances

Problem	Planning weeks	Employees	Hours with demand	Time step
P1-8w-9e	8	9	14.5	15 minutes
P2-10w-6e	10	6	8.5	30 minutes
P3-4w-45e	4	45	10.5	15 minutes
P4-12w-8e	12	8	9.5	15 minutes
P5-6w-15e	6	15	14.5	30 minutes
P6-12w-10e	12	10	24	30 minutes
P7-12w-18e	12	18	14.5	30 minutes
P8-2w-13e	2	13	24	1 hour
P9-12w-18e	12	18	15.5	30 minutes

alizing the demand structure of a representative day in the problem. As can be seen, for all problems except problem P3-4w-45e, that the minimum required demand coverage equals the maximum allowed demand coverage. This property makes the problems very strict with respect to coverage of demand. Another observation is that the problem instances differ in terms of how the demand develops throughout a day. For problems like P1-8w-9e, P3-4w-45e, P5-8w-15e and P7-12w-18e, the demand structure is identified by sudden and frequent changes in demand. The opposite is true for problems like P2-10w-6e and P4-12w-8e.

Table 8.2 is providing information regarding the number of constraints and variables generated when applying the MIP model using the shifts created by the SDH. Problems like P3-4w-45e, P6-12w-10e, P7-12w-18e and especially P9-12w-18e stand out in terms of both the number of constraints and the numbers of variables. The values presented in Table 8.2 provide increased insight into the size of the search space for each problem, but it is still hard to conclude the complexity of the problems.

8.2 Parameter Values

The parameters used when designing shifts in the SDH, reducing the shifts with the SR extension and rostering the employees using the MIP model and the PALNS, are presented in Table 8.3. These parameters apply to the problem instances described in Section 8.1. The parameters for desired shift durations and the limit on consecutive days is already declared in the mathematical formulation of the SDH-SR and MIP model, while the remaining parameters are used indirectly either in the SDH or when creating different sets used in both the MIP model and the PALNS. Some of these parameters are described in more detail below.

The parameter *time defining shift day* represents the time for when a shift is considered to belong to the next calendar day. The value 24:00 indicates that shifts starting later than

Table 8.2: Resulting number of constraints and variables when running MIP model

Problem	Number of constraints	Number of variables
P1-8w-9e	79 116	51 794
P2-10w-6e	24 300	19 070
P3-4w-45e	140 238	107 816
P4-12w-8e	64 688	41 002
P5-6w-15e	70 269	55 541
P6-12w-10e	183 110	149 736
P7-12w-18e	227 790	146 612
P8-2w-13e	16 699	13 282
P9-12w-18e	263 556	176 600

24:00 belong to the next day, which is equivalent to say that a shift belongs to the calendar day the shift starts. If the parameter value, on the other hand, is 20:00, this would imply that a shift starting at 22:00 is defined to belong to the next calendar day. Another parameter that needs to be addressed is the *allowed weekly rest duration*. This parameter is used to define which off-shifts are created by the SDH, thoroughly discussed in Chapter 5. To ensure as tight constraints as possible, Big M is equal to the number of employees.

Even though the parameter values presented in Table 8.3 apply to all of the problem instances, there are a few exceptions. In some of the problem instances, certain parameters are explicitly given other values than what is presented in Table 8.3. All such exceptions are listed in Table 8.4 below.

8.3 Weight Parameters

In this section, the weights used both in the SR extension, and for the MIP model and PALNS, are presented.

8.3.1 Weight Parameters of SR extension

For the proposed SR extension, the weights presented in Table 8.5 are applied. In some cases, the generation of shifts with undesired durations is strictly necessary to enable feasible schedules. The user is therefore enabled to specify the allowed shift duration for shifts created by the SDH, which may differ from what constitutes a desired shift duration. Applying shifts of undesired duration should be avoided unless strictly necessary, and the weights W^L and W^S are thus set accordingly. Note also that there is made no difference between positive and negative demand deviation, neither between the use of short and long shifts.

Table 8.3: Parameters used for both shift design and rostering

Explanation	Parameter	Value
Allowed shift durations [hours]		[5, 12]
Desired shift durations [hours]	$[\underline{V}^S, \overline{V}^S]$	[6, 12]
Default contracted hours for a full time employee		40
Time defining shift day [hours]		24:00
Default daily rest requirement [hours]		8
Default weekly rest requirement [hours]		36
Allowed weekly rest duration [hours]		[36, 90]
Maximum rewarded weekly rest [hours]	$\overline{V}^R$	72
Default employee specific day offset		04:00
Default number of preferences per week		[1, 3]
Default duration of each preference [hours]		[4, 8]
Desired limit of consecutive days	L^{CD}	5
Big M for mapping use of shift	M	Calculated

8.3.2 Weight Parameters of the MIP model and the PALNS

For the MIP model and the PALNS, the weights presented in Table 8.6 are used. These weights represent the intentional weights set by the user, aiming to translate the user's intentions and priorities into weights that can be interpreted by the two two rostering methods.

As it appears in both Chapter 4 and 6, the objective of the MSP could be regarded as twofold. The first aim is to minimize the deviation between actual and ideal demand coverage, while the second aim is to maximize the perceived fairness of the schedules. To enable the users to control how the two implemented rostering solution methods behave regarding these two objectives, we propose a novel weighting approach. In this approach, the different fairness aspects, highlighted in light grey in Table 8.6, are translated into costs in hourly demand deviation. In other words, the user is enabled to state how many hours of demand deviation that the rostering methods should accept in order to satisfy a certain fairness aspect. To exemplify, the weight of eight for one partial weekend represents that the user is willing to accept up to eight hours of demand deviation in order to prevent an employee from working a partial weekend. If the demand deviation becomes nine hours, the desire of satisfying demand should surpass the desire of avoiding partial weekends, and a partial weekend should be assigned. Hence, the user can rate the different fairness aspects relative to each other while at the same time determining the relative weight between fairness and demand deviation.

Table 8.4: Exceptions in parameter values

Parameter	P4-12w-8e	P6-12w-10e	P9-12w-18e
Time defining shift day	20:00	-	-
Daily rest requirement	-	8 or 9*	11
Default employee specific offset	-	-	10:00

* Value is employee-specific, i. e. one employee can have 9 and another 8.

Table 8.5: Weights used in the SR extension

Explanation	Weight	Value
Excess demand deviation	W^{D+}	1
Deficit demand deviation	W^{D-}	1
Use of long shifts	W^{L}	5
Use of short shifts	W^{S}	5

All the intentional weights are set based on discussions with Medvind and our impression of what is commonly practiced in real-life organizations. As seen in Table 8.6, there is made no difference between positive and negative demand deviation, i.e., over- and understaffing. The weight of the least favored fairness score, representing the obtained fairness score for the least favored employee, is set to 0.1. This means that the individual fairness score is weighted 10% of the collective fairness score. This is in line with the weighting used in Rönnberg and Larsson (2010).

In addition to the weights presented in Table 8.6, a penalty weight for infeasibility is included in the calculations of the objective function of the PALNS, denoted W^{IP}. The infeasibility weight adjusts to what degree infeasibility should affect the incumbent solution. The weight and its associated value are presented below. The value is set based on preliminary testing.

$$W^{IP} = 12$$

Adjustments of the Intentional Weights

As discussed in Chapter 6, the intentional weights are slightly modified in order to adjust the weights based on problem-specific characteristics. As the actual contribution of the intentional weights depends on problem-specific characteristics, like the time step or number of employees, the implemented rostering methods could yield different results on problems with different characteristics, but where the intention of the user is the same.

Table 8.6: Values of all intentional weights used in the MIP model and the PALNS

Weighting aspect	Corresponding weight	Value
Excess demand deviation	W^{D+}	1
Deficit demand deviation	W^{D-}	1
Least favoured fairness score	W^{F}	0.1
One hour additional weekly rest	W^{R}	0.5
One less working hour	W^{B}	1
One partial weekend	W^{PW}	8
One isolated working day	W^{IW}	10
One isolated off day	W^{IO}	10
One violation of consecutive working days	W^{CD}	12
One preference granted	W^{P}	5

This issue is, for instance, seen in the relation between the accumulated fairness and the least favored fairness score. The least favored fairness score is always represented by a single score, while the magnitude of the accumulated fairness score depends on the number of employees. In cases with many employees, the accumulated fairness will become higher relative to the least favored employee compared to cases with few employees. Thus the various adjusting factors are introduced, in order to preserve the intention of the weights regardless of the characteristics of the problem at hand. How the various adjusting factors are set, is presented below.

The A^D parameter is used to scale the weight of positive and negative demand deviation to represent hourly demand deviation. The A^D parameter is thus defined as:

$$A^D = \text{time step}$$

The A^F parameter is used to scale the weighting of the lowest fairness score relative to the total accumulated fairness score so that the relation of these two fairness aspects remain intact regardless of the number of employees. The A^F parameter is defined as:

$$A^F = \text{number of employees}$$

Note that the lowest fairness score is multiplied with the number of employees, instead of dividing the accumulated fairness score by the same number. Dividing the accumulated

fairness score, could in cases with low fairness scores, result in objective values closer to zero, which would influence the sensitivity of the rostering methods.

The A_e^B parameter is used to scale the weighting of deviation in contracted hours, as it is desirable that one hour deviation in contractual hours should mean more for an employee with few contractual hours than for an employee with many contractual hours. Therefore, the weights regarding contracted hours are tailored to each employee by calculating the A_e^B as below:

$$A_e^B = \frac{\text{contracted hours for full time employment}}{\text{contracted hours of employee } e}$$

Contracted hours for full-time employment represents the default contracted hours of an employee working full time. This default value is a parameter value set to 40 hours per week in Section 8.2.

The A^P parameter scale the weighting of preferences in order to represent hourly preferences, as preferences are originally stated for time periods equal to the time step. Remember also that the preferences are normalized based on the total amount of preferences stated for the whole planning period. The preference value P_{et} thus depends on the length of the planning horizon, which needs to be accounted for. Thus, the A^P parameter is calculated as:

$$A^P = \text{time step} \cdot \text{number of weeks in planning horizon}$$

8.3.3 Specific Parameters of the PALNS

This section presents parallelization parameters, as well as ALNS-related parameters and parameters used in Record to Record Travel.

All parameters related to parallelization are presented in Table 8.7. The sharing start parameter represents at what time the subprocesses should start to share solutions. We start sharing solution after 60 seconds, as preliminary tests have shown that this gives a good balance between diversification and co-operation between the subprocesses. The sharing frequency, on the other hand, determines how often the subprocesses should share solutions. We use a sharing frequency of every 10 seconds as a default. It is commonly more conventional to base theses types of parameters on the number of iterations and not on time as we have done. The latter approach is, however, selected for two reasons. First, the time-based approach makes sure that every subprocess shares solutions synchronously. This is beneficial as the probability of a subprocess sharing a solution with itself is smaller compared to an iteration-based sharing. Secondly, it makes sure that a balance between

diversification and intensification can be fixed, regardless of the size of the problem instance. An iteration-based sharing could lead to situations where the sharing would occur very frequently, or almost never, neither of which is beneficial. Finally, the number of subprocesses affects how many subprocesses should be used in parallel. The default value is all available CPU-cores. A higher value would mean that one core would have to alternate its work between two different subprocesses, and this will likely lead to performance degradation. Not using all cores would mean that some cores are idle, likely meaning that one does not utilize the hardware to its fullest potential.

Table 8.7: Values of parallelization-related parameters

Parameter	Value
Sharing start	After 60 seconds
Sharing frequency	Every 10 seconds
Number of subprocesses	Number of available cores

The rewards and penalties related to new solutions are an essential part of the adaptiveness of the ALNS-framework. The parameters and their corresponding values are listed in Table 8.8. Operators that improves on the current solutions are rewarded with ψ_{better}. If the solution also improves on the best-found solution, the associated operators are instead rewarded with ψ_{best}. This ensures that the heuristic will select these operators with a high probability in future iterations. To make sure that the search is diversified and that the heuristic has the ability to break free from local optima, the accepted solutions are also rewarded with $\psi_{accepted}$. If the operator pair produces a rejected solution, the penalty $\psi_{rejected}$ is applied, making the probability smaller for selecting these operators going forward. The decay parameter η is used to balance the weight of historical performance versus the latest result. A high decay value means that historical performance is more important than recent results. These values are inspired by (Santini, 2019), and are adjusted based on preliminary tests. The intention is to make the PALNS adapt to using well-performing operators more frequently while also retain the ability to use less successful ones to broaden the search.

Hill Climbing is used as the standard acceptance criteria, while Record to Record Travel is only used to investigate the effects of the acceptance criteria. As a consequence, no default value for has been chosen for the start threshold. We use 0 as an end threshold as Santini et al. (2018) found this value consistently gave the best results for their PALNS-implementation. Because of time constraints, we have not been able to implement a robust method to set the step length. We set the step length in such a way that the threshold reaches 0 after around 75% of the iterations. We use the results from preliminary tests to find the number of iterations per problem instance. We acknowledge that setting a parameter after gaining insights in the instances is not a robust method. As we are only concerned with investigating the potential Record to Record Travel might have and are not using it when analyzing the PALNS or comparing it against the MIP model, we find that it will suffice. A more rigorous approach would be to run the heuristic for a short period,

Table 8.8: Parameters related to adaptiveness

Aspect	Corresponding parameter	Value
Decay	η	0.9
Reward for best sol	ψ_{best}	10
Reward for better sol	ψ_{better}	4
Reward for accepted sol	$\psi_{accepted}$	2
Penalty for rejected sol	$\psi_{rejected}$	0.7

and use the number of iterations per time unit to estimate the number of iterations for the entire runtime, and use this number to calculate the step length. Table 8.9 summarizes the parameters and their values.

Table 8.9: Parameters of Record to Record Travel

Parameter	Value
Start threshold	No default
End threshold	0
Step length	Calculated

8.3.4 Operator Weight Parameters

The PALNS utilizes a high number of different destroy and repair operators, introduced in Chapter 7, that are able to explore various neighborhoods. To guide the search in the direction believed to be most beneficial, a number of weights are included in the scoring functions used by some of the operators. Preliminary testing indicated a higher performance of operators when the search direction is based on weights different from those used in the objective function. This is due to the fact that repair operators, combined with local search operators, seem to perform better in some neighborhoods than others. This is caused by the way they are implemented in addition to the combination of repair operators with local search operators.

Destroy Operator Weights

The destroy operator weights are used in the scoring function that destroys either the worst employee or the worst week in the current solution. The weights with their corresponding values are presented in Table 8.10. The values of the weights presented in the table is set based on preliminary testing.

Table 8.10: Values of weights used in destroy operator scoring function

Weighting aspect	Corresponding weight	Value
Violations of Isolated Off Days	W^{EIO}	10
Violations of Isolated Working Days	W^{EIW}	10
Violations of Consecutive Days	W^{ECD}	1
Violations of Partial Weekends	W^{EPW}	8
Violations of hard constraints	W^{HC}	10
Deviation from ideal demand	W^{DID}	10

Repair Operator Weights

Similar to the weights for the destroy operators, the repair operator weights are used in different scoring functions applied in different repair operators. Table 8.11 displays these weights and their corresponding values. The weights are set based on preliminary testing. Weights highlighted in gray correspond to hard violating hard constraints. The remaining weights correspond to violations of soft constraints. The value of the weights is connected to the importance of avoiding violations of this type, where higher values indicate a preference for not violating the corresponding constraint. Additionally, in some cases, the implemented local search operators are highly efficient at resolving violations, allowing for lower weight values.

Table 8.11: Values of weights used in repair operator scoring functions

Weighting aspect	Corresponding weight	Value
Violation of Daily Rest	W^{VDR}	500
Violation of Weekly Rest	W^{VWR}	200
Violation of Isolated Off Days	W^{VIO}	20
Violation of Isolated Working Days	W^{VIW}	20
Violations of Consecutive Days	W^{VCD}	40
Violations of Partial Weekends	W^{VPW}	50

Chapter 9

Computational Study

This chapter presents the proposed solution methods to the MSP are presented and discussed. The test environment is presented in Section 9.1, while test configurations are stated in Section 9.2. In Section 9.4, technical aspects of the PALNS are analyzed in further detail. In Section 9.3, the results of both the Shift Design Heuristic (SDH) and the proposed Shift Reduction (SR) extension are analyzed. Finally, Section 9.5 compares the performance of the PALNS heuristic to the MIP model presented in Chapter 6.

9.1 Test Environment

All tests are performed on Solstorm, a High Performance Computing cluster belonging to the Department of Industrial Economics and Technology Management. To ensure comparability, we are performing all computations on the same server rack. The hardware and software specifications are displayed in Table 9.1.

Table 9.1: Specification of hardware and software used in testing

Processor	4 x Intel E5-2670v3
Cores / Frequency	48 cores / 2.3GHz
Memory	64Gb
Operating System	CentOS Linux 7
Python version	3.7.4
Gurobi version	9.0.2

Both the Gurobi optimization solver and the PALNS are able to utilize all of the available cores to run more computations in parallel. The solver utilizes parallelization to work

on different branches of the branch and bound tree simultaneously. As a result, the solver should work efficiently on large and well-balanced search trees (Glockner, n.d.), as this is a prerequisite to keep all worker processes active and useful. The PALNS, on the other hand, ensures that each core consistently performs work by alternating between sharing solutions and searching independently. Although the approaches differ in strategies, we still think it is a fair and useful comparison to make, as both algorithms exploit their inherent strengths and try to overcome their limitations.

9.2 Test Configuration

This Section briefly elaborates on the configurations used for testing. First of all, the parameters and weights presented in Chapter 8 are used for all runs. Furthermore, Hill Climbing is used as the acceptance criteria for the PALNS. In addition to these specific configurations, some additional aspects are discussed in more detail.

A central aspect of the MSP is to obtain scheduling solutions quickly. Scheduling managers often consider it impractical to wait for exact solutions, which often requires a lot of time to obtain. To account for this preference, the solution approaches has a restricted runtime of 15 minutes (900 seconds). In some cases it is interesting to compare the PALNS against the MIP with increased runtime. This applies to situations where the emphasis is turned more towards the quality of a solution or where the MIP is unable to obtain a feasible solution within 15 minutes. The importance of runtime is thus less prominent. In these situtations the MIP is run for five hours (18 000 seconds). All gap values are calculated against the best bounds obtained after running the MIP model for 5 hours.

In the following analysis, all nine real-life problems instanced are tested. For some of the more in-depth analysis, however, only a subset of the test instances are used. This is done in order to reduce the comprehensiveness of the analysis. In such cases, only the problems P3-4w-45e, P5-6w-15e, P6-12w-10e, P7-12w-18e and P9-12w-18e are tested. As indicated in Chapter 8, these problems seem to be of greater complexity and proved to be the most challenging problems to solve in preliminary tests. For a solution approach to be applicable to the MSP it is of paramount importance that it can handle large and complex problem instances. These problems are thus regarded as the more interesting to evaluate in-depth.

9.3 Analysing the Shift Design Heuristic

This section aims to analyze the effects of the SDH presented in Chapter 5. A key research question in this master's thesis is to examine the effect of addressing the shift design problem and designing heuristic shifts tailored to individual problems. The analysis of the SDH is performed in three stages. First, the effects of the proposed SR extension is analysed in Subsection 9.3.1. In Subsection 9.3.2, the performance of the SDH is then compared against the manually created shifts provided by Medvind. Finally, in Subsection 9.3.3, the SDH is compared to the implicitly generated shifts from the IMP-MIP model, as the

implicitly generated shifts are assumed to be ideal.

9.3.1 Effects of SR Extension

As discussed in Chapter 5, the SR extension is offered as an extension to the SDH, aiming to minimize the number of possible candidate shifts. This is done in order to reduce the complexity of solving the rostering stage in the MSP. However, the reduction in possible shifts provides fewer scheduling alternatives and thus reduced flexibility. It is therefore relevant to analyze how the SR extension affects the complexity of solving the various problem instances, and how the quality of the solutions are influenced. First, we elaborate on the test results regarding the MIP model before concluding on the results regarding the PALNS.

Effects of SR Extension on MIP Model

Table 9.2 presents the results on running the nine real-life problems with the MIP model both with and without the SR extension. Note that a light grey color is used to distinguish what is being compared more easily, and this layout is pervasive for the remainder of this section. Note also that results for the MIP model with the SR extension applied are given in percentage improvement or worsening compared to the MIP model without the SR extension. The runtime limit is set to five hours.

Table 9.2: Test results for MIP model with and without the SR extension

Problem	Without SR extension				With SR extension				
Name	Best bound	Objective value	Optimality gap	Runtime [sec]	Shift reduction	Best bound	Objective value	Optimality gap	Runtime [s]
P1-8w-9e	2 589.10	2 589.10	Optimal	3 643	-13.85%	-0.01%	-0.01%	Optimal	1 965
P2-10w-6e	1 994.22	1 994.22	Optimal	67	-48.61%	-0.21%	-0.21%	Optimal	51
P3-4w-45e	6 595.36	6 531.06	0.98%	18 000	-67.45%	-1.19%	-0.60%	0.39%	18 000
P4-12w-8e	2 177.95	2 177.95	Optimal	735	-48.54%	-0.90%	-0.90%	Optimal	515
P5-6w-15e	3 514.83	3 513.65	0.03%	18 000	0.00%	0.00%	0.00%	0.03%	18 000
P6-12w-10e	3 904.90	3 104.33	25.79%	18 000	-35.71%	-0.09%	+12.89%	11.33%	18 000
P7-12w-18e	6 588.68	5 849.56	12.64%	18 000	-27.71%	-1.28%	+1.35%	9.18%	18 000
P8-2w-13e	1 043.23	1 043.23	Optimal	465	-46.73%	-4.06%	-4.06%	Optimal	140
P9-12w-18e	7 885.46	3 532.05	123.25%	18 000	-40.00%	-0.25%	-0.52%	123.86%	18 000

When evaluating the results, it is evident that for all problem instances, the best bound is higher in the case without the SR extension applied. The only exception is for problem P5-12w-8e, where the best bound is identical for the two cases. This is, however, expected, as the shift reduction is 0%, indicating that no shifts are removed. The indication of a higher best bound for the MIP model without shift reduction supports the idea that multiple shift alternatives facilitate increased flexibility and greater freedom to fulfill the various fairness aspects. However, attention must be paid to the fact that the best bound is a theoretical consideration. The best bound has little practical significance, as the scheduling manager is only able to make use of obtained solutions. This underlines the importance of addressing

the complexity of solving problems, and to evaluate the actual objective values achieved within the set time frame.

In Appendix C, Figure C.9 and Figure C.10 are provided in order to visualize how the quality of the solutions obtained with the MIP model progress with time, both with and without the SR extension. Based on the figures, it is evident that the MIP model with the SR extension obtains better solutions faster. This is true for all plotted test instances, except for P1-8w-9e, where the opposite holds. The tendency that the SR extension is enabling the MIP model to obtain better solutions faster is particularly evident in cases where the MIP model struggles with finding high-quality solutions. For problems like P3-4w-45, P6-12w-10e and P7-12w-18e, the difference in gap relative to runtime is quite significant. For P6-12w-10e, the SR extension is also providing the first feasible solution much faster than without the SR extension. These findings are important, as obtaining quality solutions fast is of great practical value.

When it comes to the quality of the schedules, the objective value achieved within the time limit of five hours is evaluated. For all test instances reaching the optimality condition, Table 9.2 shows that the objective value is the highest for the MIP model without shift reduction. This is expected based on the same argumentation used to explain a higher expected best bound. The difference is, however, small, and in most cases well below 1%. In the cases where no optimal solution is achieved, on the other hand, it is interesting to see that the MIP model for some problems obtain a higher objective value when the SR extension is applied. This is the case for the problems P6-12w-10e and P7-12w-18e. This is in line with the trend visualized in Figure C.9 and C.10, indicating that the MIP model with shift reduction tends to provide better solutions faster than without shift reduction.

A more detailed insight into the quality of the schedules is provided in Table 9.3. The table specifies how the MIP model relates to the two quality measures, demand deviation and scheduling fairness, and is based on the results provided in Table 9.2. The demand deviation is given as average hours of demand deviation per week. As can be seen, there is no difference in achieved demand deviation for the MIP model with and without shift reduction, except for a minor difference in problem P3-4w-45e. When it comes to the obtained fairness, it can be seen that the average fairness and the lowest fairness score is higher for the problems obtaining a better objective value. It is, however, interesting to see that the relative difference between the average fairness score and lowest fairness score remains relatively consistent regardless of whether the SR extension is applied to the MIP model. The conclusion is thus that the SR extension looks to have an impact on the obtained fairness level, but more or less, no significance for the demand coverage.

Based on the performed analysis, it is reasonable to assume that the SDH on a general basis operates best with the SR extension. As seen, the difference in achieved schedules seems to be negligible. With an increased performance in generating quality schedules fast, the SR extension is thus believed to perform better in practical applications. This conclusion serves as the backdrop for only running the MIP model with the SR extension in the remainder of this chapter. In the following, the use of SDH with the SR extension applied is hereafter referred to as the SDH-SR.

Table 9.3: Increased insight into the quality of the MIP model solutions

Problem	Without SR extension			With SR extension		
Name	Demand deviation	Average fairness score	Lowest fairness score	Demand deviation	Average fairness score	Lowest fairness score
P1-8w-9e	1.50	263.43	255.83	1.50	263.40	255.86
P2-10w-6e	0	303.82	285.46	0	303.40	282.72
P3-4w-45e	92.06	139.68	136.35	93.13	139.00	135.34
P4-12w-8e	1.00	248.93	248.16	1.00	246.70	245.96
P5-6w-15e	0	213.05	211.94	0	213.05	211.94
P6-12w-10e	0	282.55	278.87	0	318.69	317.57
P7-12w-18e	4.00	298.80	275.09	4.00	303.75	269.37
P8-2w-13e	0	73.16	70.93	0	70.23	67.64
P9-12w-18e	5.00	183.24	146.50	5.00	187.31	95.57

Effects of SR Extension on PALNS

The results of running the PALNS with and without the SR extension are presented in Table 9.4. These tests are run for 15 minutes. When evaluating the results, it can be seen that the PALNS with reduced shifts, in general, provides the best solutions. For seven of the nine problems, the PALNS with the SR extension applied provides the best solutions. As the PALNS is more thoroughly reviewed in Section 9.4 and 9.5, a more detailed analysis of the PALNS is not provided in this section. We settle with the conclusion that the PALNS provides the best results when the SR extension is applied. This is used as an argument for only evaluating the PALNS with the SR extension in the remainder of this chapter.

Table 9.4: Test results for PALNS with and without the SR extension

Problem	Without SR extension	With SR extension		
Name	Objective value	Shift reduction	Objective value	Difference in objective value
P1-8w-9e	2 437.93	-13.85%	2 438.11	+ 0.01%
P2-10w-6e	1 934.32	-48.61%	1 949.03	+ 0.76%
P3-4w-45e	5 873.75	-67.45%	5 854.86	- 0.32%
P4-12w-8e	2 148.92	-48.54%	2 151.24	+ 0.11%
P5-6w-15e	3 227.02	0.00%	3 271.76	+ 1.39%
P6-12w-10e	3 516.73	-35.71%	3 525.40	+ 0.25%
P7-12w-18e	5 646.77	-27.71%	5 867.03	+ 3.91%
P8-2w-13e	970.58	-46.73%	965.37	- 0.54%
P9-12w-18e	5 947.85	-40.00%	5 984.15	+ 0.61%

9.3.2 Comparing SDH-SR to Manually Created Shifts

It is interesting to evaluate how the SDH-SR performs relative to the manually created shifts provided by Medvind. In the analysis, only the MIP model is applied. This is done in order to reduce the comprehensiveness of the analysis. Table 9.5 presents the results of running the MIP model with the SDH-SR and with the manually created shifts, where the time limit is restricted to five hours. Identical shifts represent how many of the shifts generated by the SDH-SR that is found in the manually created shifts. Created shifts, on the other hand, represent the total amount of candidate shifts created by either the SDH-SR or manually for the entire planning period.

Table 9.5: Test results of manually created shifts compared to the shifts designed by the SDH with SR extension

Problem	Identical shifts	MIP model with SDH-SR					MIP model with manually created shifts				
Name		Best bound	Objective value	Optimality gap	Runtime [s]	Created shifts	Best bound	Objective value	Optimality gap	Runtime [s]	Created shifts
P1-8w-9e	94.20%	2 588.96	2 588.96	Optimal	1 965	224	2 709.44	2 290.43	18.29%	18 000	1 459
P2-10w-6e	65.41%	1 990.03	1 990.03	Optimal	51	185	Infeasible	-	-	-	774
P3-4w-45e	94.53%	6 517.02	6 491.72	0.39%	18 000	138	-	-	-	18 000	677
P4-12w-8e	100.00%	2 158.34	2 158.34	Optimal	515	88	2 248.31	2 248.31	Optimal	13 911	340
P5-6w-15e	95.56%	3 514.83	3 513.65	0.03%	18 000	270	Infeasible	-	-	-	590
P6-12w-10e	0.19%	3 901.53	3 504.43	11.33%	18 000	540	3 843.68	3 695.60	4.01%	18 000	759
P7-12w-18e	68.26%	6 504.59	5 928.37	9.18%	18 000	334	-	-	-	18 000	1 010
P8-2w-13e	43.86%	1 000.95	1 000.95	Optimal	140	57	Infeasible	-	-	-	144
P9-12w-18e	23.81%	7 865.65	3 513.57	123.86%	18 000	504	-	-	-	18 000	2 355

The first observation made when analyzing Table 9.5 is that the MIP model with manually created shifts only obtains a solution for three of the nine problem instances. This is compared to nine out of nine for the MIP model with SDH-SR. Additionally, only one optimal solution is found, compared to the four optimal solutions found with SDH-SR. These results could partially be explained due to the increased amount of available shifts. For all problem instances, the amount of manually created shifts is considerably higher compared to the number of shifts designed by the SDH-SR. Recall that in the MIP model, shifts are characterized by their exact starting time throughout the planning period. Two equivalent shifts, but applied on two different days, are thus regarded as separated and unique shifts. The increased amount of manually created shifts is mainly because these shifts are not tailored to each specific day. This means, for instance, that shifts intentionally designed for weekends will be a possible shift alternative also for the weekdays, even though weekend shifts might be poorly adapted to the demand structure for weekdays. The MIP model, thus have to evaluate a larger set of shifts, where several of the shifts are unlikely to be used. This is an inefficiency that looks to influence the test results in Table 9.5, as the MIP model with manually created shifts appears to use longer time, and in several cases do not reach a feasible solution.

Another significant result, revealed by Table 9.5, is that three of the problems, P2-10w-6e, P5-6w-15e and P8-2w-13e, are infeasible in the case where the manually created shifts are used. This could, of course, be due to errors in the shift data provided by Medvind, but illustrates the fact that manually created shifts could result in an infeasible problem.

Based on the analysis above, it is reasonable to conclude that the shifts created by the SDH-SR provide the most robust set of shifts. The results show that the MIP model performs better on general scheduling problems when applying the SDH-SR, compared to applying the manually created shifts. This is an argument that emphasizes the importance of addressing the shift design problem and the contribution of the SDH in solving the MSP. It is, however, interesting to observe that in the cases where the MIP model with manually created shifts provide a solution, the best bound and/or the objective value is higher compared to the solutions obtained with the SDH-SR. This is an interesting result, as one important aim with the SDH is to facilitate as high-quality schedules as possible. In the following analysis, we will try to identify why the manually created shifts, for some of the problems, seem to provide better schedules.

Table 9.6 provides a more detailed insight into how the obtained solutions relate to demand deviation and employee fairness. Note that the achieved demand deviation is equal, regardless of whether the SDH-SR or manually created shifts are applied. There is, however, a difference in achieved fairness.

Table 9.6: Increased insight into the quality of manually created shifts compared to the shifts designed by SDH-SR

Problem	MIP model with SDH-SR			MIP model with manually created shifts		
Name	Demand deviation	Average fairness score	Lowest fairness score	Demand deviation	Average fairness score	Lowest fairness score
P1-8w-9e	1.50	263.43	255.83	1.50	234.34	214.85
P4-12w-8e	1.00	246.70	245.96	1.00	256.92	256.14
P6-12w-10e	0	318.69	317.57	0	355.99	335.67

For problem instance P1-8w-9e, Table 9.5 shows that the best bound with manually created shifts is the highest, however, as the optimality gap is larger there is some uncertainty in this results. The MIP model with the SDH-SR does, however, provide the best solution obtained within the time limit, which explains the better results in Table 9.6. For problem P6-12w-10e, on the other hand, the best bound is highest when applying the SDH-SR. The manually created shifts do, however, provide a better solution within the time limit, even though the MIP model is given a larger set of shifts to evaluate. This is an interesting result. Problem P6-12w-10e entails a 24-hour demand. The increased performance of the manually created shifts for this problem could indicate that the SDH-SR is not performing ideally for problems with demand spanning 24 hours. Finally, problem P4-12w-6e provides superior results, even though 100.00% of the shifts designed by the SDH-SR is found in the set of manually created shifts. This could indicate that the SDH-SR is unable

to generate favorable shifts. The latter cases will be analyzed in greater detail, with the intention of identifying possible areas of improvement for the SDH-SR.

When analyzing which shifts that are applied in the two solutions for problem P4-12w-10e, it becomes evident the shifts used in the optimal schedule for manually created shifts are the exact same shifts that are used in the optimal schedule when applying the SDH-SR. The performance of the SDH-SR in terms of constructing work shifts can thus not be the explanation for the difference in obtained quality. By studying the solutions more closely, it turns out that the differences in quality lie solely in the performance on weekly rest. When evaluating the total amount of weekly rest hours for the entire planning horizon, and for all employees, the manually created shifts obtain 64.25 additional weekly rest hours compared to the SDH-SR. This is in line with the weaknesses of the SDH in terms of generating weekly off-shifts, discussed in Chapter 5. Additionally, the SDH creates weekly off-shifts based on the available shifts. Since the manually created set of work shifts are larger than the set of work shifts generated by the SDH-SR, the set of possible off-shifts is thus also larger in the case of manually created shifts. This gives the MIP model increased flexibility in terms of selecting weekly off-shifts. These aspects are the likely explanations for the different quality achieved.

When analyzing the problem P6-12w-10e, the selected shift combination in the two obtained solutions differ. A visualization of the structure of the shifts for both the manually created shifts and the SDH is presented in Figure 9.1. The shift structure is presented for a representative weekday. The upper shifts are those used when applying the SDH-SR, while the blue dashed line represents transition times. The orange dashed lines, on the other hand, represent times when a shift is starting or ending, that is not a transition time.

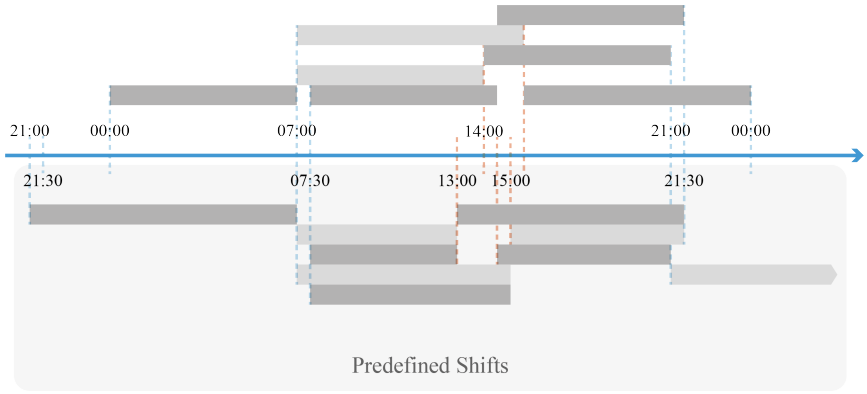

Figure 9.1: Shift structure of SDH-SR (upper) and manually created shifts (lower) for a representative day in problem P6-12w-10e

The first observation is that none of the shifts are identical when comparing the SDH-SR shifts to the manually created shift. In addition, none of the shifts used when applying the SR extension are overlapping calendar days. This is, however, the case for the manually

created shifts, where a shift spanning from the evening the day before is applied. In a realistic setting, it is reasonable to believe that the manually created shift spanning from 21:00 to 07:00 would be preferred to the shift generated by the SDH-SR spanning from 00:00 to 07:00. Whether this could explain the inferior performance of the SDH-SR in terms of quality is challenging to say, but illustrates that our simplified approach to handling demand for 24 hours is possible to improve. It is, however, important to stress that shifts that overlap calendar days are enabled by the SDH, but looks to be removed when applying the SR extension. It is also interesting to see that the manually created shifts are structured differently from the SDH-SR for the long and unchanging demand between 07:30 and 21:00. Also here it is hard to say whether the difference is the backdrop for the different performance, but it is evident that it would be interesting to explore more advanced handling of long and unchanging demand periods in the SDH.

9.3.3 Comparing SDH-SR to Implicit Shifts

The IMP-MIP model assigns employees to work by allocating them to a starting time with a set work duration. The model is free to choose how the employees are allocated, as long as the duration is within set limits and general scheduling rules are satisfied. The combination of the starting time and assigned duration is regarded as an implicit shift. As these shifts are assumed to be optimal for the MSP, they provide an interesting benchmark to compare the SDH-SR against.

In order to test the IMP-MIP model, special test instances are introduced. These test instances are identical to the corresponding test problems presented in Chapter 8, except that they are reduced to a one-week planning horizon. This modification is introduced in order to enable the IMP-MIP model to achieve feasible solutions. The modification does not provide an ideal basis for comparison, as only evaluating one week do not represent realistic problems and could restrain some of the strengths of the IMP-MIP model. However, it is assumed that the comparison still yields representative results, as neither fairness aspects nor coverage of demand should be affected by this problem modification. In Table 9.7, the identical shifts indicate how many of the shifts created by the SDH-SR is also generated by the IMP-MIP model. Table 9.7 presents the results of running the MIP model using the SDH-SR, compared to running the IMP-MIP model, with a time limit of five hours. Created shifts indicate the total amount of shifts created by either the SDH-SR or the IMP-MIP model for the one-week planning horizon.

The implicitly generated shifts provide both a better solution and a higher best bound for all test instances. This is also evident by evaluating Table 9.8, providing increased insight into the performance regarding demand deviation and employee fairness. It is thus clear that the IMP-MIP model generates better shifts than the SDH-SR. However, attention must be paid to the cost of generating these shifts. As seen in Table 9.7, the IMP-MIP model uses a considerably longer time obtaining solutions. In addition, a solution is only obtained for four of the nine problems. The shifts created by the SDH-SR could still be regarded as superior to the implicitly generated shifts in terms of real-life applicability. It is, however, interesting to analyze how the implicit shifts differ from the SDH, to identify how the SDH can be improved.

Table 9.7: Test results for MIP model with SDH-SR and for the IMP-MIP model on modified test instances

Problem	Identical shifts	MIP model with SDH-SR					IMP-MIP model with implicit shifts				
Name		Best bound	Objective value	Optimality gap	Runtime [sec]	Created shifts	Best bound	Objective value	Optimality gap	Runtime [sec]	Created shifts
$P2_{IMP}$	61.11%	197.04	197.04	Optimal	0	18	202.02	202.02	Optimal	197	21
$P4_{IMP}$	100.00%	152.05	152.05	Optimal	0	8	194.09	194.09	Optimal	3 314	8
$P8_{IMP}$	48.28%	499.90	499.90	Optimal	121	29	529.29	529.29	Optimal	3 514	44

Table 9.8: Increased insight into the quality of the implicit shifts compared to the shifts designed by the SDH-SR

Problem	MIP model with SDH-SR			IMP-MIP model with implicit shifts		
Name	Demand deviation	Average fairness score	Lowest fairness score	Demand deviation	Average fairness score	Lowest fairness score
$P2_{IMP}$	0	30.04	28.00	0	30.77	29.00
$P4_{IMP}$	0	17.55	14.52	0	22.39	18.73
$P8_{IMP}$	0	35.44	30.14	0	37.22	34.92

As the SDH is designed in order to imitate the shifts designed by the IMP-MIP model, the number of identical shifts are interesting. For $P4_{IMP}$, the SDH-SR generates all the shifts that are generated by the IMP-MIP model. This result is compelling, even though it could be argued that this would be expected based on the simple demand structure of P4, which is found in Appendix B. Note also that despite using the same shifts, the IMP-MIP model obtains better results. When inspecting the solution closer, the IMP-MIP model performs better on the maximization of weekly rest. This is the same conclusion as for problem P4-12w-8e with manually created shifts, underlining the potential of improving the part of the SDH generating the off-shifts. As the number of identical shifts for $P2_{IMP}$ and $P8_{IMP}$ are lower, it is interesting to analyze these problem instances closer.

The shift structure for the SDH-SR and the IMP-MIP model for $P2_{IMP}$ is visualized in Figure 9.2 for a representative day. Despite the simple demand structure, none of the shifts are identical. It is interesting to observe that, based on how work shifts are generated in the SDH, both of the shifts generated by the IMP-MIP model is generated by the SDH but removed by the SR extension. This emphasizes the consequences of applying the SR extension, and indicates that the performance of the SDH with respect to creating identical shifts as the IMP-MIP model might be better than indicated in Table 9.7.

For problem $P8_{IMP}$, the different shift structures are visualized in Figure 9.3. Note the orange dotted lines, indicating start and end times for shifts that are not a transition time. One shift is generated both by the SDH-SR and the IMP-MIP model and is colored green.

As seen, the IMP-MIP model generates more shifts variation than what is generated by

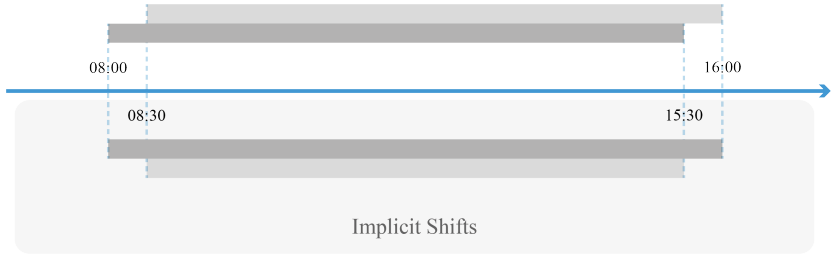

Figure 9.2: Shift structure of SDH-SR (upper) and implicit shifts (lower) for a representative day in problem P2$_{IMP}$

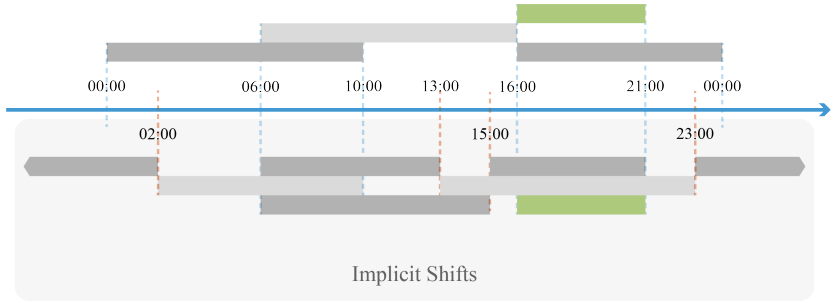

Figure 9.3: Shift structure of SDH-SR (upper) and implicit shifts (lower) for a representative day in problem P8$_{IMP}$

the SDH-SR. Also, the SDH-SR does not create shifts that overlap calendar days, which the implicit shifts do. This result was also seen when analyzing the shifts generated by the SDH-SR for the 24-hour demand in P6-12w-10e. Another interesting observation is that the IMP-MIP model seems to generate some shifts that do not start and/or end in transition times. This deviates from the overall trend revealed in our preliminary work, where the implicit model almost without exception did generate shifts that started or ended in transition times. Such a result is particularly interesting as the SDH is based on how the IMP-MIP model generates its shifts. This supports the idea that there may exist underlying structures that make more sophisticated handling of 24-hour shifts interesting to explore further.

9.4 Technical Analysis of the PALNS

This section aims to analyze PALNS-specific results and choices. The general aspects of the PALNS are discussed in Subsection 9.4.1. As parallelization is a relatively unexplored theme in the literature, we find this part of the PALNS particularly interesting to evaluate and are analyzed in more detail in Subsection 9.4.2. Finally, the acceptance criteria Hill Climbing is compared to Record to Record Travel in Subsection 7.1.4.

9.4.1 General Aspects

Some of the aspects that the PALNS shares with the conventional ALNS framework are presented in this subsection. The effects of randomness is discussed before results regarding speed and stagnation are analyzed. The adaptiveness of the heuristic is then elaborated on.

Uncertainty and Randomness

Randomness comes into play in several parts of the PALNS, most notably in the selection and usage of operators. In the proposed heuristic the destroy operators can destroy up to 45% of the solution at random. Additionally, the repair operators have some random elements as well. The selection of operators is also affected by randomness, as we utilize roulette wheel selection. Another random element, which is not found in conventional ALNS, is the sharing of solutions. The sharing of solutions is not deterministic, as the order of which the subprocesses reaches this sharing event is dependent on the iteration times, which in turn dependents on the operators.

The effect of randomness on the solution has been investigated by running each problem instance ten times, with different random seeds for each run. Table 9.9 shows the corresponding results. One standard deviation constitutes less than 1% of the average objective value for each test instance, except for problem P9-12w-18e where it consitutes 1.41%. Even though we have not benchmarked the results against comparable heuristics, we find the variance to be low. The results indicate that the heuristic is robust in its ability to provide solutions and that the results are not due to any inherent variance in the solution approach. In all comparisons throughout this section, the average objective value over the ten runs for a given problem instance is used. When plotting the gaps, a representative run is used, that is, the test run with the objective value closest to the average value.

Table 9.9: Results for PALNS

Problem	Average objective value	Standard deviation
P1-8w-9e	2 438.11	18.60
P2-10w-6e	1 949.03	7.72
P3-4w-45e	5 854.86	45.52
P4-12w-8e	2 151.24	13.06
P5-6w-15e	3 263.17	19.32
P6-12w-10e	3 552.98	26.24
P7-12w-18e	5 945.70	50.80
P8-2w-13e	965.37	5.73
P9-12w-18e	5 949.67	84.16

As the variance is within satisfactory limits for all problem instances, only five test runs are performed for additional test runs. This includes tests for modified problem instances or tweaks in the heuristic. The variance is assumed to be similar for these cases as well, and the five runs should be sufficient to handle the inherent uncertainty in the PALNS.

Speed and Stagnation

Figure 9.4 and 9.5 shows the gap for each problem instance as a function of time. Note that the plot includes the construction time. For most of the problems, the construction time is negligible, but for a few instances, the construction time is around 20 seconds, and in the most extreme case, it reaches 62 seconds. For solutions to have a real-life value, the gap should be well below 25%. To make sure that the plots both show the general trends and provides the necessary detail, only gaps below 25% are shown. This means that solutions are regularly found before they appear on the plot. This consideration applies to all plots in this section.

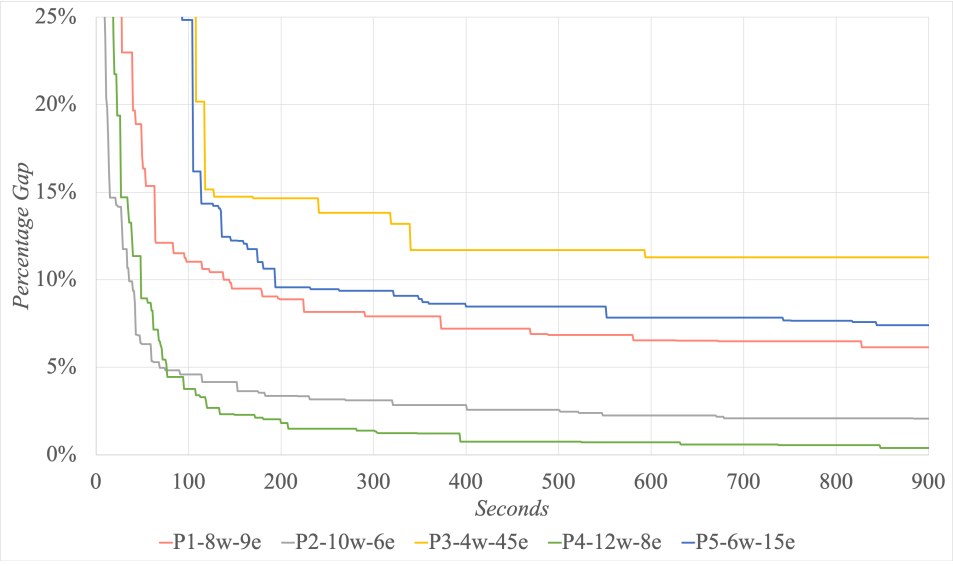

Figure 9.4: Development of PALNS solutions for P1-8w-9e, P2-10w-6e, P3-4w-45e, P4-12w-8e and P5-6w-15e

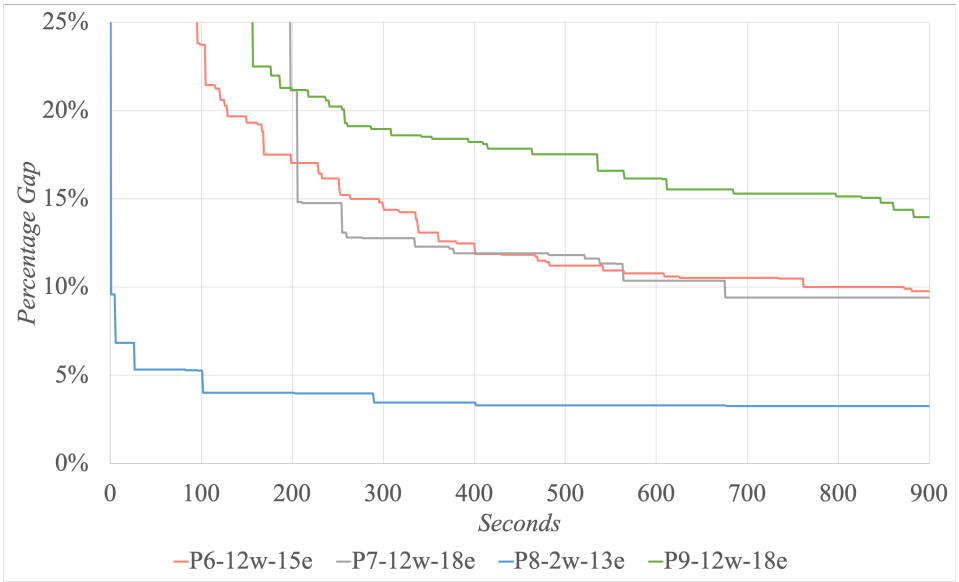

Figure 9.5: Development of PALNS solutions P6-12w-10e, P7-12w-18e, P8-2w-13e and P9-12w-18e

As can be expected from a heuristic, the PALNS quickly improves on the constructed solutions but have more trouble finding better solutions as the current solution gets closer to the optimal solution. Within 300 seconds, the larger part of the improvements has already been reached for most of the problems. The PALNS only achieves relatively small improvements for the remainder of the runtime. Problem instances like P1-8w-9e, P2-10w-6e and P4-12w-8e seem to reach a plateau early. Based on the results of the MIP model present in Section 9.3, these problems could be regarded as less complex. The opposite holds for problems like P6-12w-10e, P7-12w-18e and P9-12w-18e, which in Figure 9.4 seems to require more time to reach a plateau. This might indicate that the heuristic could find even better solutions if allowed to run longer for these instances.

When the interval between improving solutions increases, while the magnitude of each improvement decreases, it might indicate that the PALNS has stagnated. Close to optimal solutions, heuristics will naturally stagnate, as it becomes more challenging to improve on the current solution. Excluding the problem instances reaching an optimality gap of less than 5%, problem instances like P1-8w-9e, P3-4w-45e and P5-6w-15e seem to have the highest degree of stagnation. With default parameters, the week operators will destroy one out four weeks for P3-4w-45e, and one out six weeks for P5-6w-15e. For problem P1-8w-9e, four out of the nine employees will be destroyed by some of the employee operators. These destroy sizes are some of the largest sizes across all problem instances and might be a factor behind the stagnation. Because such a large part of the solution is deleted and repaired each iteration, the heuristic is quick to improve on the initial solution. When small changes are needed, the PALNS still destroys large parts of the solution, making it difficult to achieve the necessary precision to improve upon the current solution.

Adaptiveness

The adaptiveness of the ALNS framework is meant to select the currently best performing operators with the highest frequency. By actively rewarding operators that improve on the current solution, an ALNS should continuously adapt itself to the search space. The adaptiveness should enable the inclusion of a higher number of operators compared to a regular Large Neighborhood Search. The subprocesses in the PALNS does not share any information regarding operator performances, and will thus adapt independently of the other subprocesses. To measure the effectiveness of the adaptiveness, all problem instances are compared to a tweaked heuristic where the adaptive behavior is disabled. The resulting heuristic will be more similar to a parallel Large Neighborhood Search than a parallel ALNS. Going forward the tweaked heuristic will be denoted as the PLNS.

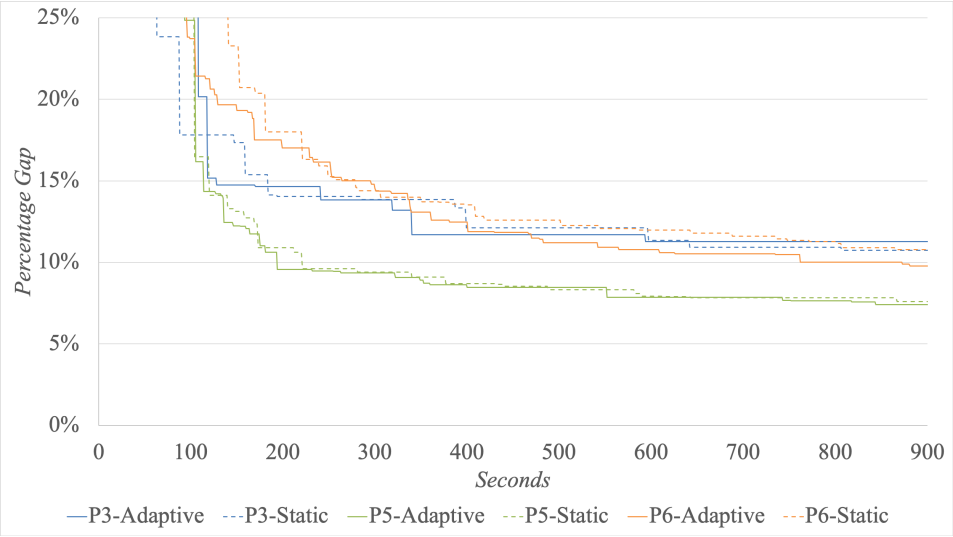

Figure 9.6: Development of PALNS and PLNS for P3-4w-45e, P5-6w-15e and P6-12w-10e

The development of PALNS and PLNS for the five selected problems are shown in Figure 9.6 and 9.7. The PALNS gives the best results for the majority of the problem instances, both in regard to speed and objective value. In some cases the difference is substantial, like the objective values for the problem P6-12w-10e and P7-12w-18e, indicating that the adaptiveness provides a significant benefit for these instances. However, the PALNS is only slighter better for problems P5-6w-15e and P9-12w-18e. For P3-4w-45e, the PLNS finds the best solution, but the difference is relatively small for this instance as well. These are some of the problems with the lowest frequency between new improvements, indicating that the PALNS has stagnated. The stagnation might indicate that the PALNS does not have any operators that are able to find new and better solutions. As this situation lasts over many iterations the updating of weights will result in a uniform distribution for selecting the operators, essentially converting the PALNS to a PLNS. To break this stagnation the

heuristic relies more on randomness in the pairing of operators or within the operators themselves. In situations like these, the contribution of adaptiveness is insignificant, and the PALNS and PLNS should perform equally well. In general, the effect of the adaptiveness seems relatively small. A likely explanation is that the benefits of using multiple subprocesses, each taking independent choices, reduces the significance of selecting the most suited operators.

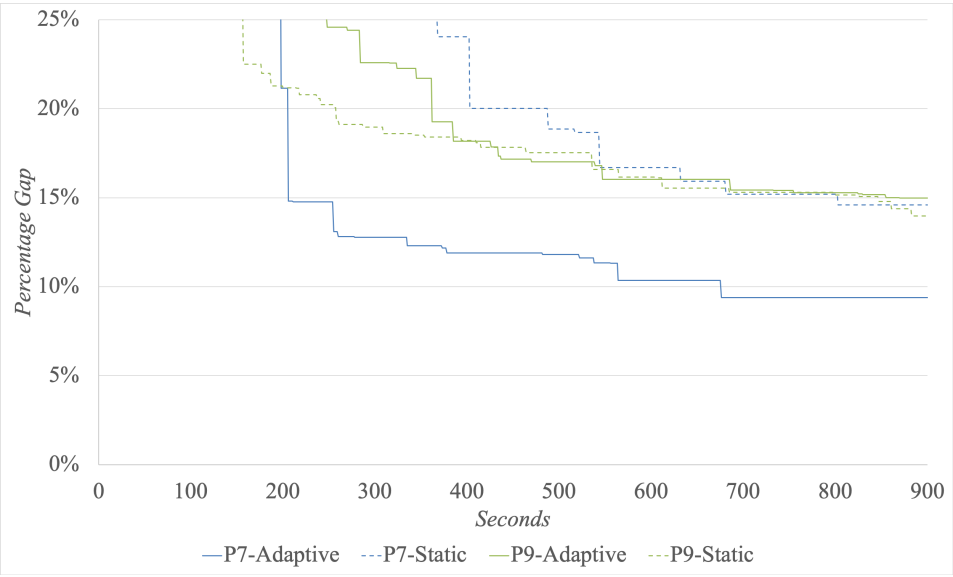

Figure 9.7: Development of PALNS and PLNS for P7-12w-18e and P9-12w-18e

9.4.2 The Effects of Parallelization

This section investigates the effects of parallelization and how it differs from a conventional ALNS. The effects of collaboration between subprocesses are briefly discussed, before analyzing how the number of subprocesses affects the PALNS.

Number of Subprocesses

To investigate how the number of subprocesses affects the performance of the PALNS, a wide range of parameter values have been tested. As a Solstorm node consists of 48 cores, we have used 48 subprocesses as the default configuration. Note that using only one subprocess is equivalent to a conventional ALNS.

Figure 9.8 shows the relationship between the number of subprocesses and the total number of iterations. The plot clearly shows diminishing returns. As the number of subprocesses increases, the marginal gain in the total number of iteration decreases. As the num-

ber of subprocesses goes towards the number of available cores, the marginal gain seems to approach zero. These results correspond with Amdahl's law, presented in Chapter 3, which shows that even a small serial fraction in an algorithm will significantly affect the expected speedup. Although the use of a shared queue should be a highly efficient method to share states, it still might result in a non-negligible serial fraction of the code, along with parallelization overhead in general. Another factor might be the actual acceptance of new candidates. It seems reasonable to believe that the frequency of accepted solutions is positively correlated with the number of subprocesses, as this entails a more diverse search. Each accepted solution incurs significant overhead as this leads to a copying of the solution state, which is among the slowest operations in the heuristic. An implication is that the largest gains in parallelization are achieved when scaling from a basis of a few subprocesses. Adding additional subprocesses is not likely to improve upon our current configuration with 48 subprocesses. Furthermore, these results indicate that one has to consider the trade-off between a more diversified and intensified search. On the one hand, many subprocesses will increase the diversification as additional subprocesses will provide variation in the search. On the other hand, a smaller number of subprocesses will lead to a higher number of iterations per subprocess, and thus the search will be more intensified.

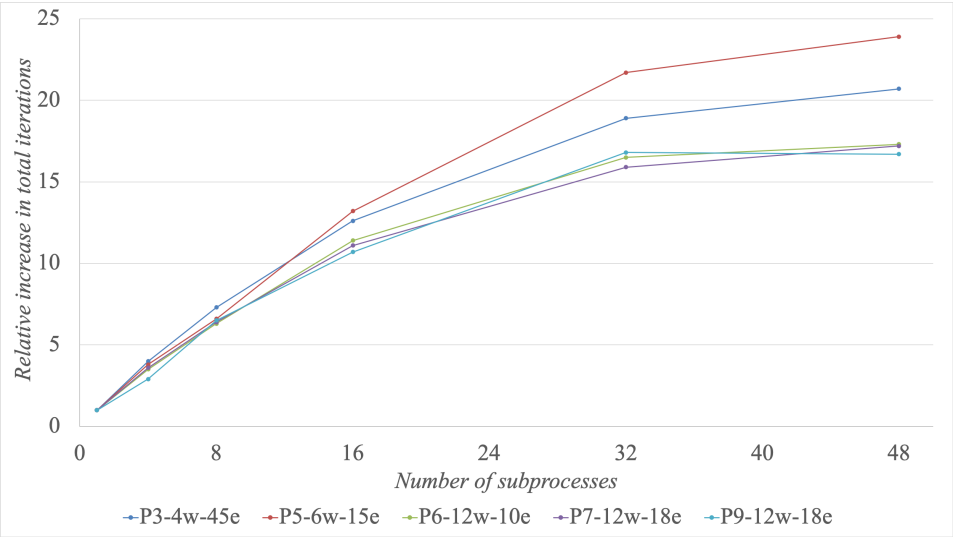

Figure 9.8: Relative increase in total iterations for 4, 8, 16, 32 and 48 subprocesses compared to 1 subprocess

Figure 9.9 shows the gaps as a function of runtime for P7-12w-18e, which we find most representative for this test. The results for P3-4w-45e, P5-6w-15e, P6-12w-15e and P9-12-w-18e can be found in C.1.1. The results show that the number of subprocesses can have a large impact on the performance of the PALNS. Using only one subprocess is dominated by every other configuration. The single subprocess configuration finds inferior solutions

and is significantly slower than the rest. This supports the idea that the PALNS is an improvement over the conventional ALNS.

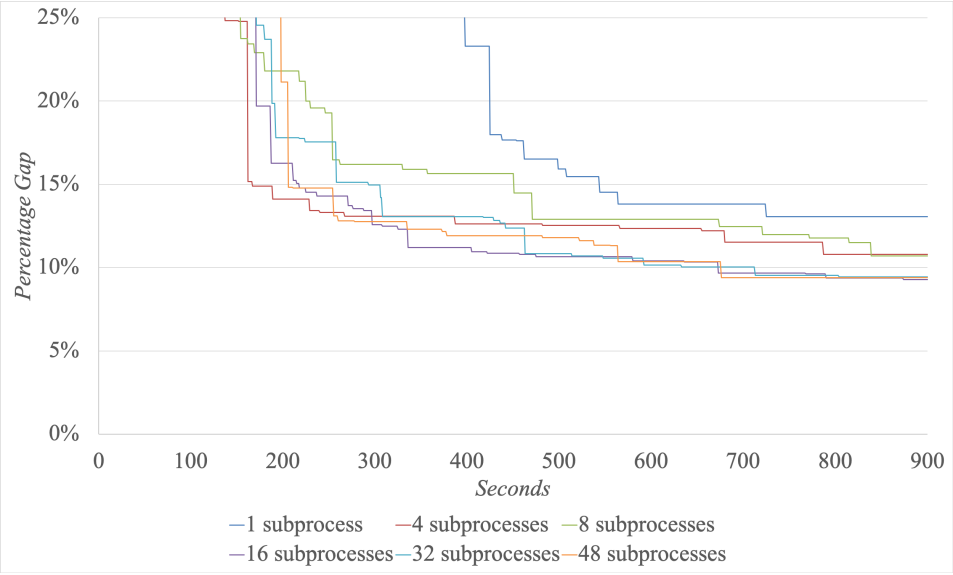

Figure 9.9: Development for different number of subprocesses for P7-12w-18e

It seems like configurations with more subprocesses are slower to find solutions below a 25% gap than solutions with fewer subprocesses. The exception to this is the case with only one subprocess. This could be because each subprocess performs fewer iterations between the sharing intervals. A consequence of this is that a given subprocess is less likely to find an improving solution compared to a configuration with a higher number of iterations per subprocesses. The configurations with more subprocesses seem to make up for the slow start gradually and surpassing the other configuration before the search is concluded. The most likely explanation is that the effects of sharing are more significant towards the end of the search. Furthermore, such configurations seem less prone to stagnation, which can be contributed to the increased diversification.

The configurations with 16 or more subprocesses seem to yield the best results. They consistently perform average or better. None of these configurations seems to dominate each other. Although the natural choice of utilizing all available cores yields good results, the results indicate that other configurations perform just as well. Accordingly, two conclusions can be drawn: the number of subprocesses can significantly affect how the heuristic performs, and it is better to use at least 16 subprocesses for our test environment.

The Effects of Sharing

To investigate the benefits of sharing solutions, the PALNS, with 48 subprocesses, has been compared to a modified heuristic, where no sharing occurs during the search. This modification is equivalent to running a conventional ALNS for 48 different random seeds and then choosing the best result, and is denoted non-collaborative. In essence, this modification relies solely on the variance of the problem to get better results compared to a regular ALNS. The results are shown in Figure 9.10 and 9.11.

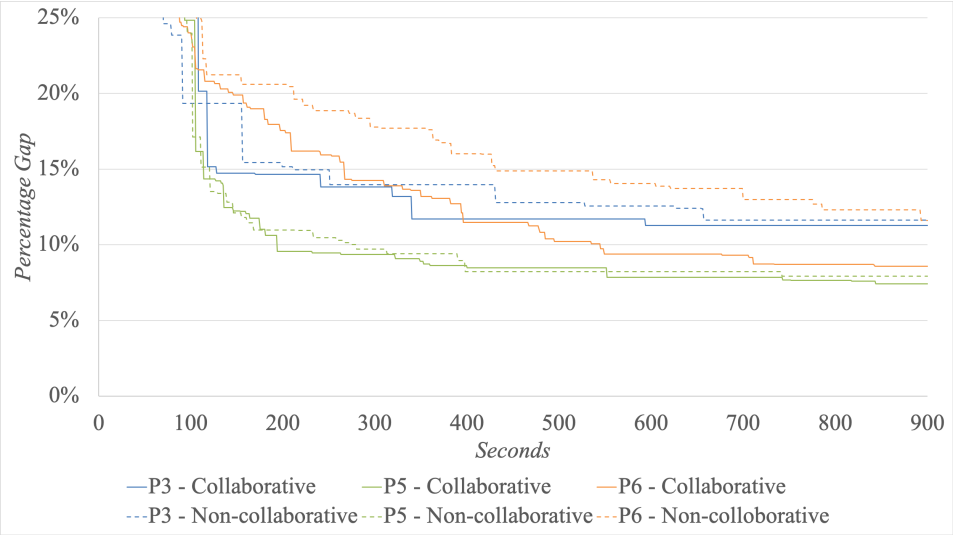

Figure 9.10: Development of collaborative and non-collaborative PALNS for P3-4w-45e, P5-6w-15e and P6-12w-10e

The PALNS performs substantially better for P6-12w-10e and P7-12w-18e. Also, it is slightly better for P3-4w-45e. For the two remaining cases the difference is negligible. Overall, this result shows that the PALNS is an improvement over the conventional ALNS. The PALNS is superior to the modified PALNS with no sharing of solutions, which in turn is at least as good as the conventional ALNS. Furthermore, changing the number of subprocesses will affect these results. As mentioned, the modified heuristic will depend on the variance to find better solutions relative to an ALNS. We expect the variance to be positively correlated to the number of subprocesses. Reducing the number of subprocesses will thus likely lead to worse solutions for the modified heuristic. This implies that some of the benefits of the PALNS will be even larger when using fewer subprocesses.

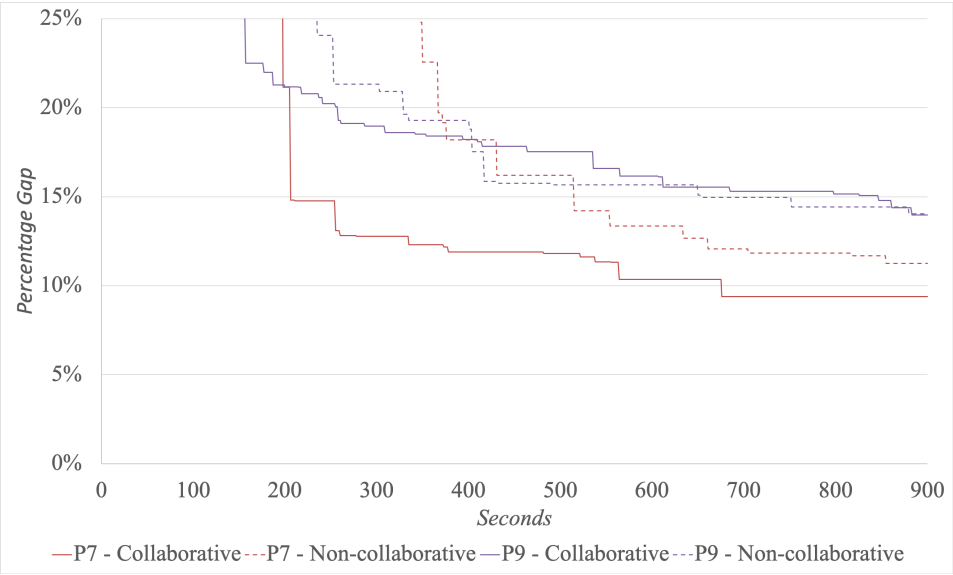

Figure 9.11: Development of collaborative and non-collaborative PALNS for P7-12w-18e and P9-12w-18e

9.4.3 Acceptance Criteria

The intention of an acceptance criterion is to control the level of diversification for the neighborhood search. Hill Climbing is usually disregarded in favor of more complex criteria for ALNS, as it is prone to get stuck in local optima and does not provide any diversification. To explore if this applies to the PALNS as well, the Hill Climbing criterion is compared against the Record to Record Travel criterion. Five different configurations of Record to Record Travel has been tested. The result for P9-12w-18e, which we find the most representative, is shown in Figure 9.12. The remaining results are provided in Appendix C.1.2.

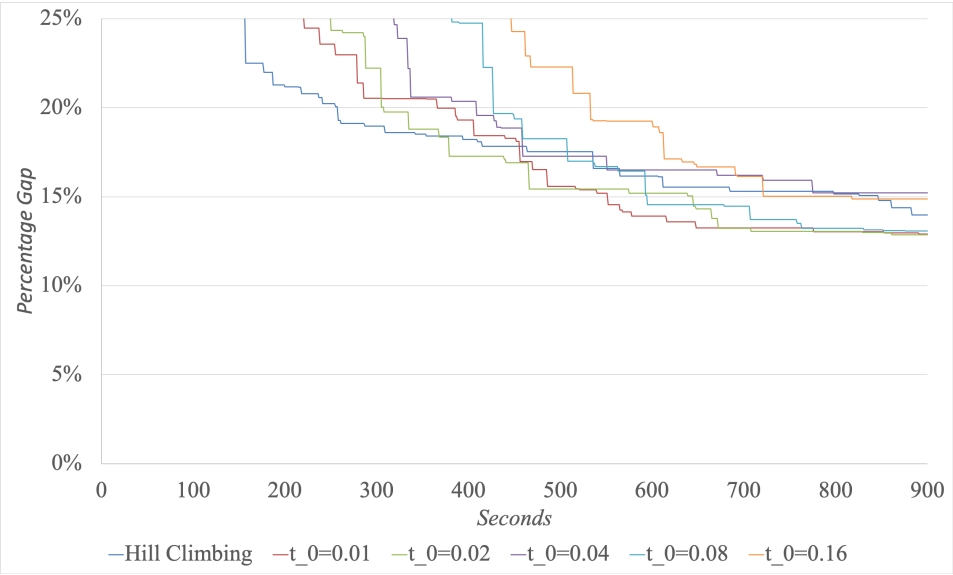

Figure 9.12: Development of Hill Climbing and Record to Record Travel with threshold of 1%, 2%, 4%, 8% and 16% for P9-12w-18e

Generally, Hill Climbing tends to improve on the current solution faster than Record to Record Travel. This should come as a no surprise, as the greedy behavior should be more efficient when the heuristic manages to find many new feasible solutions, as is typically the case in the early phase of the search. What is surprising, however, is how well Hill Climbing performs throughout the entire search. For the conventional ALNS, the additional diversification of Record to Record Travel is expected to provide significantly better results. This is clearly not the case for the PALNS. The Record to Record Travel with a threshold of 2% seems like the most robust configuration. Problem P7-12w-18e is the only instance where it is not among the best configurations, for the remaining four it yields among the best results. Hill Climbing and Record to Record Travel with a 1% threshold seems to outperform the remaining Record to Record configuration. It is especially interesting to see that Hill Climbing is performing as well as the best Record to Record Travel configurations for P3-4w-45e and P5-6w-15e, as stagnation seems to be a significant issue for these instances. A likely explanation is that the parallelization, in combination with large destroy sizes and the usage of randomness in the algorithm, already provides sufficient diversification for the search.

9.5 Evaluating the Implemented Solution Approaches for the Medvind Scheduling Problem

This section is devoted to the computational results regarding the performance of the PALNS, presented in Chapter 7 and the implemented MIP model, presented in Chapter 6. A comparison between the two solution approaches is made on their performance on all

the problem instances. In addition, the applicability of the two approaches is evaluated in regard to the MSP.

In Subsection 9.5.1, the MIP model and and the PALNS is compared with a restricted runtime of 15 minutes. A comparison is then made between the two solution approaches on their suitability to the provided problem instances. Furthermore, to gain insight into the overall performance of the PALNS, its results are compared to the best bound of the implemented MIP model with a runtime of five hours, obtained in Section 9.3. Additionally, an in-depth qualitative analysis of the two solution methods in regard to the fairness aspects is completed. Subsection 9.5.2 evaluate the scalability of the PALNS and the MIP model.

9.5.1 PALNS Performance on Real Problem Instances

All real-life problem instances are analyzed in order to assess the performance of the PALNS and the implemented MIP model on different complexities associated with different industries.

Comparing the PALNS to MIP Model to Real-Life Instances

The results for the PALNS and the MIP model are presented in Table 9.10. The table includes both the results for the MIP model with 15 minute runtime and five hour runtime to compare the performance of the PALNS on similar criteria, as well as the overall performance of the PALNS, in regard to optimality.

The results show that the PALNS is able to produce satisfactory results for most of the problem instances when compared to the MIP model. The largest reduction in performance for the PALNS compared to the MIP model is observed on P3-4w-45e, with a difference of -7.17%. This result is assumed to be caused by the inherent structure and size of the problem, in combination with the use of a large destroy sizes and design of the repair operators. For the problems where the MIP model obtains a solution, the PALNS has an average improvement over the MIP of 3.18%. It is, however, important to mention that the improvement is mainly caused by the high performance of the PALNS compared to the MIP on P6-12w-10e. Another observation is that the PALNS is able to obtain solutions relatively close to optimum on three problems where the MIP is not able to find any feasible solutions within the required 15 minutes.

The results indicate a slight advantage of the MIP model on the first four problems, in addition to P8-2w-13e, as it is able to reach optimal or close to optimal solutions. These problems appear to be less complex than the remaining four instances. This coincides with our initial assumption that the MIP would perform better on less complex problems, where it would potentially achieve optimal or close to optimal solutions within the runtime requirements. This assumption is based on the notion that the strength of a MIP model is in its ability to work systematically towards the optimal solution. Conversely, the PALNS should be more efficient at searching through the large solution space of more complex

problem instances. From the table, this is evident as the PALNS are able to find satisfactory solutions on the more complex problems within the 15 minutes runtime restriction. Problem instance P3-4w-45e is an exception. The size of the problem would indicate that it is a complex problem. However, the demand structure of the problem is less strict than the remaining problems, which could be an explanation for the results seen in the table. Studying P6-12w-10e with 15 minute runtime, the MIP achieves a solution with an optimality gap of 71.14%, while on P7-12w-18e, no feasible solution is found. The PALNS, however, is able to produce solutions that are close to or better than the solutions found by the MIP with a five hour runtime. Another interesting observation is that on the very complex problem P9-12w-18e, the PALNS achieves a gap of 29.72% within 15 minute, constituting a performance improvement from the MIP model with five hour runtime of 41.83%.

Table 9.10: Results of the MIP model and PALNS on the nine problem instances

Problem	MIP			PALNS	
Name	Best Bound $5\ Hours$	Gap $5\ Hours$	Gap $15\ Minutes$	Gap $15\ Minutes$	Improvement from MIP $15\ minutes$
P1-8w-9e	2 588.88	*optimal*	1.84%	6.02%	−4.10%
P2-10w-6e	1 990.03	*optimal*	*optimal*	2.23%	−2.23%
P3-4w-45e	6 555.34	0.39%	4.36%	11.84%	−7.17%
P4-12w-8e	2 155.48	*optimal*	*optimal*	0.05%	−0.05%
P5-6w-15e	3 514.69	0.03%	–	7.33%	–
P6-12w-10e	3 896.28	10.80%	71.14%	9.65%	35.93%
P7-12w-18e	6 504.61	9.72%	–	9.77%	–
P8-2w-13e	1 000.94	*optimal*	*optimal*	3.58%	−3.58%
P9-12w-18e	7 835.26	124.00%	–	29.72%	–

Furthermore, as observed in Table 9.10, the performance of the PALNS on P4-12w-8e is close to optimal. This indicates that the PALNS has the potential of creating close to optimal schedules, making it applicable for less complex problems as well.

Figure 9.13 shows the MIP objective values in percentage compared to the solution obtained for the PALNS, represented by the dashed orange line. The dotted vertical lines represent the point where the MIP solutions are equal to the solution achieved by the PALNS. The yellow dashed vertical line is the time restriction of 15 minutes. As the problem instances P1-8w-9e, P2-10w-6e, P4-12w-8e, and P8-2w-13e are solved to optimum by the MIP, they are not included in this graph. Additionally, to better be able to present the problem instances, some initial MIP solutions are not included in the graph.

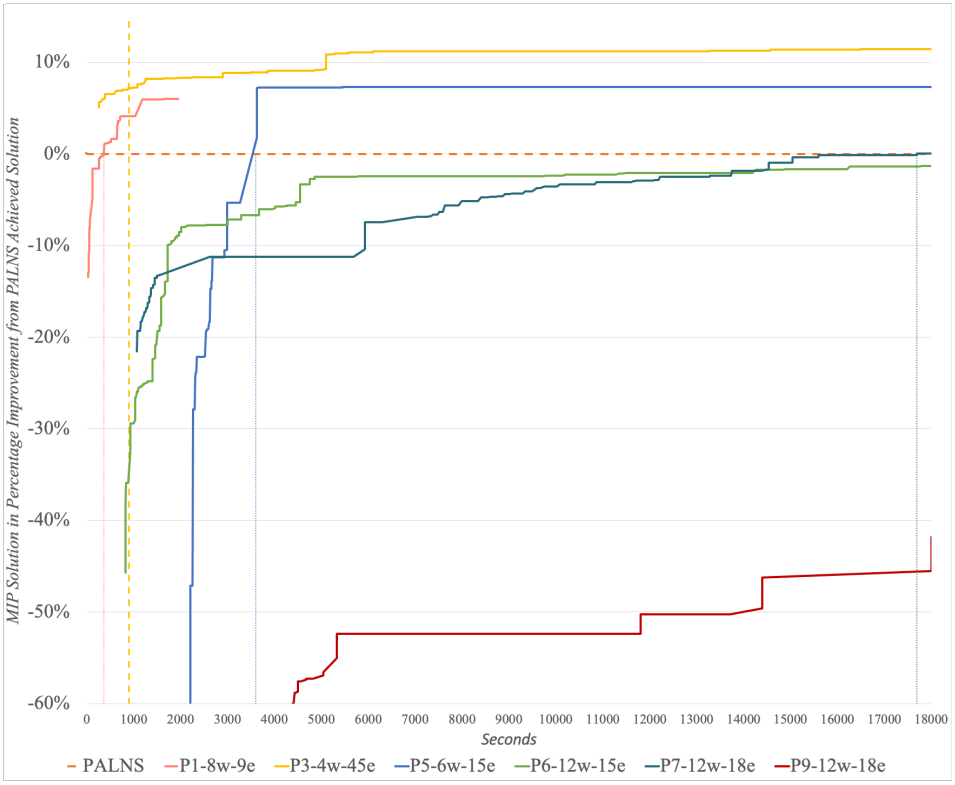

Figure 9.13: Development of MIP solutions as percentage improvement from achieved PALNS solution

The graph displays the development of the MIP over time, showing when the PALNS outperforms the MIP model for the given problem instances. This could indicate which problem complexities it would be beneficial to utilize the PALNS. As observed in Table 9.10, and further substantiated by the graph, on both P1-8w-9e and P3-4w-45e the MIP model are able to achieve better solutions than the PALNS. Studying the remaining problem instances, the MIP model is only able to find solutions that are better or equivalent to the PALNS on two problem instances; P5-6w-15e and P7-12w-18e. On P5-6w-15e, it takes the MIP 3 620 seconds to achieve a solution that surpasses the PALNS by 1.72%, while on P7-12w-18e, it requires approximately 17 700 seconds to achieve an equal solution to the PALNS.

Based on the results presented in Table 9.10 we see that the PALNS performs satisfactorily for all real-life problems. There is, however, difficult to conclude on a general problem characteristics where it is preferable to utilize one of the solutions approach over the other. While both P3-4w-45e and P9-12w-18e are large problems as shown in Chapter 8, the PALNS and MIP results differs widely. This would indicate that the size of the problem alone does not necessarily warrant the use of one solution method over the other.

The demand requirements of problem instance P3-4w-45e is relaxed, having a deviation between the minimum and maximum demand faced. This structure seems to be favored by the implemented MIP model. The PALNS, however, seem to struggle more with this structure. Conversely, on P9, where maximum and minimum demand faced are equal, the PALNS seems to have a large performance increase compared to the MIP model.

Fairness Aspects

By only considering the optimality gap produced by the PALNS on the different problem instances, it would give the impression that the PALNS is on some instances far from finding optimal solutions. However, studying the objective function and runtime in isolation makes it difficult to get an understanding of the actual schedules produced by the two solution approaches. It is, therefore, interesting to investigate how the approaches handle the fairness aspects, which have a direct impact on the employees. Additionally, it would give an indication of what causes the differences in objective value between the two approaches. Both tables presented, concerning the fairness aspects, are based on one representative run of the PALNS, instead of being the average over ten runs. This is done as it gives a more representative insight into the schedule created.

Table 9.11 presents the average hours of continuous weekly rest assigned to each employee, the number of preferences granted in percentage of total preferences, as well as the violations of the remaining fairness aspects. Each violation and granting of preference have a direct impact on the objective value. As the MIP is not able to obtain optimal solutions on most instances within 15 minutes, the results with a runtime restriction of five hours are used. The PALNS, however, is restricted to a runtime of 15 minutes.

The total deviation from contracted hours for each employee is part of the fairness score, but have been omitted in this discussion. An investigation into the obtained solutions showed no deviation on eight out of the nine problem instances presented here. P3-4w-45e showed a difference in this aspect, with only two hours over 45 employees and a planning horizon of four weeks.

The MIP model has few or no violations on problem instances where the MIP model achieves optimal solution within the runtime of five hours, in addition to problem P3-4w-45. The PALNS performs similarly, except for problem P3-4w-45e, where it has an additional 11 violations. Violations on consecutive days are particularly prominent. It is difficult to explain this behavior, as it is most likely the result of multiple factors. One explanation could be the static destroy size causing up 25% destruction on each iteration. Another explanation is that problem P3-4w-45 requires a high number of employees to cover the faced demand. If the PALNS is not able to manage the employees correctly according to the changes in demand, employees could be assigned a higher number of shifts, leading to an undesired amount of consecutive workdays. Nevertheless, having 11 violations on this problem is relatively insignificant, taking into account the number of employees and the planning horizon of the problem instance. Examining the preferences, the PALNS is able to grant a higher number of preferences. However, with the current weights, the preferences would not have a considerable impact on the objective function.

Table 9.11: Fairness violations, preferences granted and average weekly rest for the MIP model and the PALNS

Problem	MIP						PALNS					
Name	CD	PW	IOD	IWD	PG	AWR	CD	PW	IOD	IWD	PG	AWR
P1-8w-9e	0	0	0	0	71.96%	69.0	0	0	0	2	63.15%	67.1
P2-10w-6e	0	0	0	0	61.49%	70.0	0	0	1	0	54.47%	69.6
P3-4w-45e	0	0	0	0	68.13%	69.1	9	1	0	1	76.32%	64.8
P4-12w-8e	3	0	0	6	78.13%	69.8	0	3	2	6	65.82%	70.3
P5-6w-15e	0	0	0	0	83.83%	69.8	2	0	2	0	83.69%	66.5
P6-12w-10e	14	12	18	0	78.77%	67.0	2	4	12	2	62.64%	64.1
P7-12w-18e	46	10	67	0	71.13%	62.7	51	12	24	2	77.52%	59.7
P8-2w-13e	0	4	0	0	89.63%	70.7	0	4	0	0	83.28%	68.8
P9-12w-18e	80	48	106	14	70.76%	60.1	26	2	16	7	78.39%	61.8

CD = Consecutive Days, PW = Partial Weekends, IOD = Isolated Off Days, IWD = Isolated Working Days,
PG = Preferences Granted, AWR = Average Weekly Rest

Finally, we would like to point out the difference of 4.3 in average weekly rest between the two solutions for problem P3-4w-45e. The MIP model shows high performance in utilizing the employees, achieving a deviation from the maximum weekly rest of only 2.9 hours on average. The PALNS has some room for improvement, achieving a deviation from maximum weekly rest of 7.2 hours on average. Additionally, it is outperformed compared to the MIP model on weekly rest for all problems, except on problem P4-12w-8e and P9-12w-18e.

Evaluating the problems proven to be more complex for the MIP model to solve, the PALNS outperforms the MIP model with respect to undesired work patterns on all three problems. We observe a high performance on P9-12w-18e by the PALNS, compared to the MIP model. The PALNS is able to achieve 197 fewer violations of the fairness aspects while granting more preferences and giving the employees 1.7 hours additional weekly rest on average. The conspicuous explanation is the optimality gap of 124% produced by the MIP model, which substantiates the complexity of the problem.

This analysis indicates that the PALNS is struggling to achieve an optimal allocation of weekly rest. However, the maximum hours of weekly rest are fixed at 72 in our tests. By reducing the number of awarded weekly rest, the PALNS might be able to achieve solutions closer to optimum.

Table 9.12 presents the average fairness score, represented as f-value, and the lowest fairness score achieved. The lowest fairness score is included to get an understand for how well the approaches are able to distribute the assignment of shifts that have an impact on the fairness aspects, between the employees, leading to fairer solutions. By inspecting the table, we see that the f-values achieved by the PALNS and the MIP on the more complex problems are close to equal. Although the MIP violates a higher number of the fairness aspects, it is able to achieve a higher average weekly rest than the PALNS, effectively

leveling out most of the differences. Additionally, from the table it is evident that the MIP model is more efficient at distributing fairness across all employees on all problem instances, except P9-12w-18e. This behaviour by the MIP model results in a higher lowest fairness score, which on most instances is close to the average. Although the PALNS is able to avoid violations, it struggles with weekly rest and lowest fairness score. On the problem instances where the MIP outperforms the PALNS, this appears to be the explanation, as it entails lower objective values overall. As expected, less complex problems

Table 9.12: Average Fairness and Lowest Fairness Score for the MIP and the PALNS

Problem	MIP		PALNS	
Name	Average f-value	Lowest fairness score	Average f-value	Lowest fairness score
P1-8w-9e	263.40	255.86	251.55	207.85
P2-10w-6e	303.40	282.72	297.74	271.97
P3-4w-45e	139.00	135.34	127.53	111.49
P4-12w-8e	246.43	245.01	246.40	238.63
P5-6w-15e	213.05	211.94	200.18	179.79
P6-12w-10e	318.76	317.94	323.48	308.88
P7-12w-18e	303.75	269.47	305.21	263.33
P8-2w-13e	70.23	67.64	68.57	60.00
P9-12w-18e	187.31	95.57	308.97	224.51

seems to favour the exact solution method represented by the MIP model. Although the PALNS has a high performance even on small instances, it has difficulties maximizing weekly rest. On these instances the MIP accomplishes longer weekly off periods for the employees, less violations and are able to better distribute the fairness across employees. While the PALNS is not able to achieve optimality, we observe that the actual impact on the employees schedules are less significant than what is conveyed through the comparison between the MIP and the PALNS and results displayed in Table 9.10. Additionally, the results indicate an advantage for the PALNS in regard to complex problem instances. It is therefore interesting in evaluate the scalability of the PALNS on selected modified problem instances.

9.5.2 Scalability of the PALNS

To further analyze how the problem size affects the two solution approaches, additional tests are run on modified problem instances. The implemented MIP model, however, shows signs of decreasing performance as the problem size increases either with employees or weeks. In this section, the selected subset of test instances used in the previous sections is modified in regard to the number of employees and the planning horizon. These problem instances show different problem characteristics, enabling us to assess the performance of the two solution methods thoroughly.

Number of Employees

Different organizations could potentially have a widely varying number of employees. This is already present in the real-life problem instances. However, it is difficult to evaluate the effect the number of employees has on the solution method when analyzing different problem instances with a number of different characteristics. To do this, only the employee dimension is scaled, while the other dimensions are kept constant. Table 9.13 presents the selected cases and summarizes the results achieved for both the implemented MIP model and the PALNS. In addition, the results on the original instances are included and highlighted to see overall trends. It is important to notice that for some of the instances, the number of employees has both been reduced and increased, while for others the number of employees is only increased. The reason for this is the structure of the demand in the original problem instance. To keep the proportion of the employees to the faced demand approximately equivalent to the original problem, the demand faced in each demand period has to be scaled accordingly. To ease this modification, most instances have been scaled by a multiple of their original size. The exception is P3-4w-45e, which has an increase and a decrease of 15 employees.

Table 9.13: Results of the MIP model and PALNS on problem instances with modified number of employees

Problem		MIP			PALNS	
Name	Employees	Best Bound 5 $Hours$	Gap 5 $Hours$	Gap 15 $Minutes$	Gap 15 $Minutes$	Improvement from MIP 15 $Minutes$
P3-4w-30e	30	8 904.63	1.04%	4.08%	7.33%	−6.38%
P3-4w-45e	45	6 555.34	0.39%	4.36%	11.84%	−7.17%
P3-4w-60e	60	4 245.67	0.28%	7.32%	11.18%	0.00%
P5-6w-30e	30	7 046.79	0.34%	–	8.50%	–
P5-6w-15e	15	3 514.69	0.03%	–	7.33%	–
P5-6w-45e	45	1 0573.92	118%	–	10.83%	–
P6-12w-10e	10	3 896.28	10.80%	71.14%	9.65%	35.93%
P6-12w-20e	20	5 057.55	0.01%	0.02%	5.77%	−5.43%
P6-12w-30e	30	12 840.72	40.04%	–	27.04%	–
P7-12w-18e	18	6 504.61	9.72%	–	9.77%	–
P7-12w-36e	36	14 394.92	35.54%	–	28.45%	–
P7-12w-54e	54	21 624.38	29.62%	–	29.44%	–
P9-12w-9e	9	3 368.42	12.37%	372.92%	11.23%	325.16%
P9-12w-18e	18	7 835.26	124.00%	–	29.72%	–
P9-12w-36e	36	15 758.11	77.68%	–	41.33%	–

From the table, it is evident that the PALNS is efficient at solving highly complex problem instances. On eight of the ten modified problem instances studied, the PALNS shows an improvement over the MIP with a 15 minute runtime restriction. Additionally, the PALNS achieves a lower optimality gap on six of the ten modified problem instances, compared to

the MIP with a five hour runtime restriction.

Looking at P3-4w-30e and P3-4w-60e, it is interesting to notice that the PALNS achieves a lower optimality gap than produced for the original problem with 45 employees. The MIP model with a five hour runtime restriction, however, appear to perform worse when reducing the number of employees, resulting in an optimality gap of 1.04%. On the other hand, with a runtime restriction of 15 minutes, the MIP model is able to reduce the optimality gap to 4.08%. A possible explanation for the increase in performance of the PALNS when decreasing the number of employees is that a decrease in employee reduces the number of new assignments to be made in an iteration, having fewer possibilities for violations. The results for the MIP might be caused by a less fitting demand composition, making it difficult for the MIP to avoid pattern violations. The increase in performance for the MIP with a five hour runtime might be caused by slight relaxations of the problem when increasing the number of employees.

Furthermore, on the remaining modified problem instances, it is apparent that increasing the number of employees makes the PALNS more preferable, as the MIP model with a 15 minute runtime restriction are not able to find any feasible solutions on these problem instances. Additionally, comparing the results of the PALNS to the MIP model with a five hour runtime restriction shows an improvement on most of the modified problem instances, when increasing the number of employees. Interestingly, the only exceptions are P3-4w-60e, which is a modification of the problem instance P3-4w-45e, where the MIP has struggled in previous tests. This would indicate an inherent structure in the problem instance, separated from the staff size, which limits the performance of the PALNS. It has to be stated that the optimality gap for P5-6w-45e is more challenging to comprehend as the values are calculated using a negative objective value of -56 923.84 for the MIP model.

Number of Weeks

To be as general as possible, the solution approaches would have to efficiently handle both an increase and a decrease in the length of the planning period. We are therefore interested in evaluating the PALNS on problem instances where the number of weeks has been modified. The same subset of problems is modified as in the previous subsection. The modification differs between the problems. For P3-4w-45e, the number of weeks is halved and doubled. For P5-6w-15e, the number of weeks is both increased and decreased with 50%, while on the remaining problems, the number of weeks is increased and decreased with four weeks each. The results for both the MIP and the PALNS is displayed in Table 9.14. Additionally, the results of the original instances are included and highlighted.
The results show a resemblance to the results with a modified number of employees. The PALNS achieves better results in nine out of the ten scaled instances, indicating that the PALNS scale well with the increase in weeks. Studying the results of the PALNS on all instances of the P6-12w-10e shows a close to linear increase in optimality gap. This observation holds for the two modified instances of P7-12w-18e as well. The results on the modified problem instances of P9-12w-18e are the only results that show a more than linear increase in optimality gap. However, as the MIP model have a optimality gap of 150.43%, there might be that the optimal solution is much lower than the best bound

Table 9.14: Results of the MIP model and PALNS on problem instances with modified number of weeks

Problem		MIP			PALNS	
Name	Weeks	Best Bound 5 $Hours$	Gap 5 $Hours$	Gap 15 $Minutes$	Gap 15 $Minutes$	Improvement from MIP 15 $Minutes$
P3-2w-45e	2	3 265.26	0.85%	1.35%	10, 85%	−8, 57%
P3-4w-45e	45	6 555.34	0.39%	5.84%	11.84%	−7.17%
P3-8w-45e	8	12 996.88	4.76%	–	12, 77%	–
P5-4w-15e	4	2 344.48	0.03%	–	8, 32%	–
P5-6w-15e	15	3 514.69	0.03%	–	7.33%	–
P5-12w-15e	12	7 020.24	6.66%	–	10, 67%	–
P6-8w-10e	8	2 596.99	6.60%	44.63%	6, 47%	35, 84%
P6-12w-10e	10	3 896.28	10.80%	71.14%	9.65%	35.93%
P6-16w-10e	16	5343.15	20.29%	97.84%	14, 26%	73, 15%
P7-8w-18e	8	4 352.04	0.52	23.88%	5, 34%	17, 60%
P7-12w-18e	18	6 504.61	9.72%	–	9.77%	–
P7-16w-18e	16	8 725.42	16.59%	–	12, 91%	–
P9-8w-18e	8	4 516.91	0.74%	–	10, 28%	–
P9-12w-18e	18	7 835.26	124.00%	–	29.72%	–
P9-16w-18e	16	10 488.26	150.43%	–	35, 27%	–

obtained after five hours. This validates that the PALNS is able to produce attractive solutions, and is not as sensitive to the complexity of the problems as the MIP model.

Similar to other the previous results obtained, comparing the PALNS against the implemented MIP on different modifications of the problem instance P3-4w-45e, the PALNS is outperformed. Interestingly, the PALNS is not able to achieve an optimality gap below 10% for any of the modifications of P3-4w-45e. Having tested different modifications of this problem, there appears to be some inherent structure of the problem, making it difficult to achieve closer to optimal solutions for the PALNS. A possible explanation is the static destroy size, which on this problem leads to a large part of the solution being destroyed in each iteration.

Studying the modified problem instances of P9-12w-18e, it is evident that adjusting the number of weeks has an impact on the obtained MIP model solution. When reducing the number of weeks to eight, the MIP model achieves an optimality gap of only 0.74%. Additionally, the objective value is also the second to highest observed by the MIP model on any modification of P9-12w-8e. When increasing the original problem size from 12 to 16 weeks, we see a significant increase in optimality gap. The PALNS, on the other hand, is able to achieve an improvement over MIP model of 85.15% from the five our runtime restriction and a optimality gap of only 35%, calculated from the best bound found after five hours.

Chapter 10

Concluding Remarks and Further Research

The purpose of this master's thesis is two-fold. First, it should investigate the effects of heuristically generating shifts to be used in the rostering problem. To achieve this, a shift design heuristic, with an optional SR extension, has been implemented and tested against both manually created shifts and implicitly generated shifts. Secondly, the thesis explores how to design a solution method capable of achieving high-quality schedules for the MSP. A MIP model and a PALNS have been implemented and analyzed. The most important results and the main contribution of this master's thesis are concluded in Section 10.1. The chapter ends with presenting possible areas for further research in Section 10.2.

10.1 Concluding Remarks

In this section, the most important results related to the master's thesis' purpose are concluded. In order to solve the MSP in the best possible way, both the shift design problem and the rostering problem has been addressed. A Shift Design Heuristic was implemented to design favorable shifts, while a Mixed integer programming model and a Parallel Adaptive Large Neighborhood Search heuristic were implemented in order to roster the employees.

The developed SDH, aiming to solve the shift design problem, designs shifts that facilitate high-quality schedules. As the proposed SR extension provides very similar schedules as to running the SDH without it, but performing significantly better in terms of generating schedules quickly, the SDH with the SR extension is assumed to be the preferred shift design strategy. When evaluating the designed shifts against the manually created shifts, the manually created shifts were clearly outperformed. Even though the manually created shifts provided good solutions for some of the problems, they did only provide feasible solutions to three out of the nine test instances. For the six test instances not solved within the time limit of five hours, three of them were infeasible. This, in combination with the fact that manually created shifts, slowed down the process of generating scheduling solu-

tions, reinforces the contribution of the SDH developed in this thesis.

Another important aspect is the advantages offered by the SDH concerning efficiency in the shift design process. The SDH generates shifts for the entire planning horizon in less than a second, both with and without the shift reduction extension. The process of manually generating shifts is time-consuming, and the developed SDH may, therefore, provide increased practical value.

Finally, the designed shifts were tested against the ideal implicit shifts generated by the IMP-MIP model. The results indicate that the implicit shifts perform better in terms of achieved fairness for the employees. At the same time, the coverage of demand remains identical to when the shifts designed by the Shift Design Heuristic is used. The implicit shifts are expected to provide higher quality schedules, as the implicit model generates the shifts as part of the rostering process, and thus are enabled to utilize more information. However, it is interesting to see that SDH is able to generate similar shifts as generated by the IMP-MIP model, at least in cases that do not entail 24 hour demand. Additionally, the IMP-MIP model was only able to achieve feasible solutions for three of the problems within a reasonable time limit, even when the problems were reduced to one week. This point out that the IMP-MIP model has little or no practical value, and the SDH can thus be considered superior.

In regard to the rostering of employees, the results showed that the MIP model was highly efficient at solving the less complex problem instances. It was able to solve three of the nine instances to optimality while achieving close to optimal solutions on two additional problems. However, for the more complex problems, the MIP model struggled to find feasible solutions within the restricted time frame of 15 minutes. This tendency was especially prominent in the results of the scaled problem instances.

The PALNS showed promising results, achieving a feasible solution on all nine problem instances. Although the obtained solutions showed a slightly higher optimality gap for the PALNS, the difference seems to be less significant for the employees in terms of achieved fairness. When evaluating the fairness obtained for the less complex problems, the PALNS and the MIP model achieve approximately the same number of fairness violations. The MIP model was able to assign more contiguous weekly rest to the employees through better scheduling of work shifts. Additionally, the MIP model proved better at distributing the perceived fairness evenly among the employees. Through our comparison of the more complex problems where the MIP model was not able to obtain feasible solutions, the PALNS achieved satisfying results. It is evident through these results that the strength of the PALNS lies in efficiently searching a large solution space.

As for solving the MSP, the PALNS approach shows the highest potential. The MIP model is not able to handle larger problem instances, and therefore does not have the necessary applicability to solve the MSP within a reasonable time. The PALNS however, has been able to obtain solutions on all problem complexities within the required 15 minute runtime. With an additional focus on distributing fairness evenly and increasing the achieved

continuous weekly rest, the PALNS could be improved in order to obtain solutions even closer to the optimum. Furthermore, the parallelization of the heuristic has provided significant benefit when compared against a conventional ALNS, with a substantial increase in performance. In particular, the PALNS is able to perform a considerably higher number of iterations within the runtime restriction, and the search seems to be considerably more diverse compared to an ALNS.

10.2 Further Research

This section presents possible areas for further research, that has proven interesting to explore further, and where a more detailed investigation could provide added value. These areas have not been explored in detail in this master, due to lack of data and limited scope. Further research possibilities are presented thematically below.

10.2.1 Improving Shift Design Heuristic

Despite the positive results provided by the SDH, some areas for improvements are already discussed in Chapter 5, while others are revealed through the computational study in Chapter 9. First of all, the shifts created by the SDH for long demand periods with unchanging demand and days with 24 hour demand, seems to differ in structure compared to both the manually created shifts and the implicit shifts. In such cases, the shift generation algorithm could benefit from selecting start and end times for the shifts that are aligned with changes in demand. Based on our simple handling of such cases, this could be an area worth examining closer. Additionally, when comparing the SDH against the manually created shifts and the implicitly generated shifts, the two latter cases seem to facilitate improved solutions regarding the maximization of weekly rest. This indicates that the part of the SDH that generates weekly off-shifts should be a targeted area for improvements. Possible improvements have already been suggested in Section 5.3.3 and is mainly based on how the algorithm generating weekly off-shift operates. In short, only weekly off-shifts that starts when a work shift ends and that end when a work shift starts are created. The downsides of this approach are intricate, and we, therefore, refer to Section 5.3.3 for a detailed elaboration. The proposed improvement is, however, to allow off-shifts to end at times where no work shifts begin. This is assumed to influence both the complexity of allocating off-shifts, as fewer off-shifts are created, and could ensure that off-shifts facilitating maximum weekly rest are created regardless of how parameter values are selected.

10.2.2 Improving the Parallel Adaptive Large Neighborhood Search

Although the results discussed in Chapter 9 showed promising performance by the PALNS, some limitations became apparent. The heuristic showed some signs of stagnation after 300 seconds. All repair operators implemented in this thesis is based on the greedy approach or implemented as exact solution methods. Additionally, the search direction is decided by weights defined through preliminary testing. The early stagnation might, therefore, be caused by issues in the implemented operators, or a suboptimal combination of

operators. Further research into improved destroy and repair operators, are therefore encouraged.

Furthermore, the PALNS contains a large number of parameters, most of which are decided upon through preliminary tests. As this method is not as rigorous as hyperparameter tuning, and the data set is limited, further work in this area is advised. Finally, parallelization has proved to be a promising approach by taking advantage of modern hardware to a higher degree. One area that could be beneficial to explore further is to specialize the subprocesses. The problem instances could be divided into a set of subproblems. Some of the subprocesses could be applied to individual subproblems, while the remaining subprocesses is applied to the problem instances as a whole. This could potentially improve the search, as this could lead to a better balance between diversification and intensification.

Bibliography

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

Ahuja, R. K., Ö. Ergun, J. B. Orlin, and A. P. Punnen (2002). "A survey of very large-scale neighborhood search techniques". In: *Discrete Applied Mathematics* 123.1-3, pp. 75–102.

Aickelin, U. and K. A. Dowsland (2000). "Exploiting problem structure in a genetic algorithm approach to a nurse rostering problem". In: *Journal of Scheduling* 3.3, pp. 139–153.

Aickelin, U. and K. A. Dowsland (2004). "An indirect genetic algorithm for a nurse-scheduling problem". In: *Computers and Operations Research* 31.5, pp. 761–778.

Arabia, S. and H. K. Alfares (2004). "Survey , Categorization , and Comparison of Recent Tour". In: pp. 145–175.

Aykin, T. (1996). "Optimal shift scheduling with multiple break windows". In: *Management Science* 42.4, pp. 591–602.

Azaiez, M. N. and S. S. Al Sharif (Mar. 2005). "A 0-1 goal programming model for nurse scheduling". In: *Computers and Operations Research* 32.3, pp. 491–507.

Baker, K. R. (1976). "Workforce Allocation in Cyclical Scheduling Problems: A Survey". In: *Operational Research Quarterly* 27.1, pp. 155–167.

Bard, J. F. and H. W. Purnomo (July 2005). "Preference scheduling for nurses using column generation". In: *European Journal of Operational Research* 164.2, pp. 510–534.

Beckmann, F. and K. K. Klyve (2016). "Optimisation-Based Nurse Scheduling for Real-Life Instances". Doctoral dissertation. NTNU, pp. 1–127.

Bhulai, S., G. Koole, and A. Pot (2008). "Simple methods for shift scheduling in multiskill call centers". In: *Manufacturing and Service Operations Management* 10.3, pp. 411–420.

Brunner, J. O., J. F. Bard, and R. Kolisch (July 2009). "Flexible shift scheduling of physicians". In: *Health Care Management Science* 12.3, pp. 285–305.

Brunner, J. O., J. F. Bard, and R. Kolisch (2011). *Midterm scheduling of physicians with flexible shifts using branch and price*.

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

Burke, E. K. and T. Curtois (2014). "New approaches to nurse rostering benchmark instances". In: *European Journal of Operational Research* 237.1, pp. 71–81.

Burke, E. K., P. De Causmaecker, S. Petrovic, and G. Vanden Berghe (2003). "Variable Neighborhood Search for Nurse Rostering Problems". In: *IMA Journal of Management Mathematics*, pp. 153–172.

Burke, E. K., P. De Causmaecker, G. Vanden Berghe, and H. V. Landeghem (2004). *The State Of The Art Nurse Rostering*. Tech. rep., pp. 441–499.

Burke, E. K., J. Li, and R. Qu (2010). "A hybrid model of integer programming and variable neighbourhood search for highly-constrained nurse rostering problems". In: *European Journal of Operational Research* 203.2, pp. 484–493.

Burke, E., P. De Causmaecker, S. Petrovic, and G. V. Berghe (2003). "Variable Neighborhood Search for Nurse Rostering Problems". In: pp. 153–172.

De Bruecker, P., J. Van Den Bergh, J. Beliën, and E. Demeulemeester (May 2015). *Workforce planning incorporating skills: State of the art*.

De Causmaecker, P., P. Demeester, G. Vanden Berghe, and B. Verbeke (2004). "Analysis of real-world personnel scheduling problems". In: *The 5th Conference on the Practice and Theory of Automated Timetabling, PATAT 2004* November 2015, pp. 183–197.

Dean, J. S. (2008). "Staff scheduling by a genetic algorithm with a two-dimensional chromosome structure". In: *7th International Conference on the Practice and Theory of Automated Timetabling, PATAT 2008*, pp. 1–15.

Di Gaspero, L., J. Gärtner, G. Kortsarz, N. Musliu, et al. (2005). "Theory and practice of the minimum shift design problem". In: *Operations Research/ Computer Science Interfaces Series*. Ed. by G. Di Battista and U. Zwick. Vol. 32. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 159–180.

Di Gaspero, L., J. Gärtner, N. Musliu, A. Schaerf, et al. (2013). "Automated shift design and break scheduling". In: *Studies in Computational Intelligence* 505, pp. 109–127.

Ernst, A. T., H. Jiang, M. Krishnamoorthy, and D. Sier (2004). "Staff scheduling and rostering: A review of applications, methods and models". In: *European Journal of Operational Research* 153.1, pp. 3–27. URL: www.elsevier.com/locate/dsw.

Felici, G. and C. Gentile (2004). "A Polyhedral Approach for the Staff Rostering Problem". In: *Management Science* 50.3, pp. 381–393.

Gärtner, J., N. Musliu, and W. Slany (2007). "A Heuristic Based System for Generation of Shifts with Breaks". In: *Applications and Innovations in Intelligent Systems XII*, pp. 95–106.

Geitle, A. H., Ø. K. Johnsen, and H. F. E. Ruud (2019). "Heuristic Solution Methods for a Real-Life Instance of the Nurse Scheduling Problem". Doctoral dissertation.

Glockner, G. (n.d.). *Parallel and Distributed Optimization with Gurobi Optimizer*. URL: https://www.gurobi.com/pdfs/webinar-parallel-and-distributed-optimization-english.pdf.

Grano, M. L. de, D. J. Medeiros, and D. Eitel (2009). "Accommodating individual preferences in nurse scheduling via auctions and optimization". In: *Health Care Management Science* 12.3, pp. 228–242.

Grov, H., E. Kallevik, and S. Nærland (2019). *Fair Personnel Scheduling Problem with Flexible Shift Design*. Tech. rep. NTNU.

Gustafson, J. L. (1988). "Reevaluating Amdahl's law". In: 5.

Hans, E. W., M. Van Houdenhoven, and P. J. Hulshof (2012). “A framework for healthcare planning and control”. In: *International Series in Operations Research and Management Science* 168.May, pp. 303–320.

Hax, A. C. and H. C. Meal (1973). “Hierarchical Integration of Production Planning and Scheduling”.

Isken, M. W. (2004). “An Implicit Tour Scheduling Model with Applications in Healthcare”. In: *Annals of Operations Research* 128.1-4, pp. 91–109.

Jaumard, B., F. Semet, and T. Vovor (May 1998). “A generalized linear programming model for nurse scheduling”. In: *European Journal of Operational Research* 107.1, pp. 1–18.

Jenal, R., W. R. Ismail, L. C. Yeun, and A. Oughalime (2011). “A cyclical nurse schedule using goal programming”. In: *Journal of Mathematical and Fundamental Sciences* 43A.3, pp. 151–164.

Karmakar, S., S. Chakraborty, T. Chatterjee, A. Baidya, et al. (2016). “Meta-heuristics for solving nurse scheduling problem: A comparative study”. In: *Proceedings - 2016 International Conference on Advances in Computing, Communication and Automation (Fall), ICACCA 2016*.

Lapègue, T., O. Bellenguez-Morineau, and D. Prot (2013). “A constraint-based approach for the shift design personnel task scheduling problem with equity”. In: *Computers and Operations Research* 40.10, pp. 2450–2465.

Lin, C. C., J. R. Kang, D. J. Chiang, and C. L. Chen (2015). “Nurse Scheduling with Joint Normalized Shift and Day-Off Preference Satisfaction Using a Genetic Algorithm with Immigrant Scheme”. In: *International Journal of Distributed Sensor Networks* 2015. URL: http://dx.doi.org/10.1155/2015/595419.

Lü, Z. and J.-K. Hao (2012). “Adaptive neighborhood search for nurse rostering”. In: *European Journal of Operational Research* 218.3, pp. 865–876.

Marri, H. B., A. Gunasekaran, and R. J. Grieve (1998). “Computer-aided process planning: A state of art”. In: *International Journal of Advanced Manufacturing Technology* 14.4, pp. 261–268.

Moondra, L. (1976). “An LP model for work force scheduling for banks”. In: *Journal of Bank Research* 6, pp. 299–301.

Musliu, N., A. Schaerf, and W. Slany (Feb. 2004). “Local search for shift design”. In: *European Journal of Operational Research*. Vol. 153. 1, pp. 51–64.

Ni, H. and H. Abeledo (2007). “A branch-and-price approach for large-scale employee tour scheduling problems”. In: *Annals OR* 155, pp. 167–176.

Nilssen, E. J., M. Stølevik, E. L. Johnsen, and T. E. Nordlander (2011). “Multi-skill shift design for Norwegian hospitals”. In: *Journal of Applied Operational Research* 3.3, pp. 137–147. URL: www.tadbir.ca.

Ouelhadj, D., S. Martin, P. Smet, E. Ozcan, et al. (Jan. 2012). *Fairness in nurse rostering*. Tech. rep.

Pillac, V., C. Guéret, and A. L. Medaglia (2013). “A parallel matheuristic for the technician routing and scheduling problem”. In: *Optimization Letters* 7.7, pp. 1525–1535.

Pisinger, D. and S. Ropke (2007). “A general heuristic for vehicle routing problems”. In: *Computers and Operations Research* 34.8, pp. 2403–2435.

Pisinger, D. and S. Ropke (2019). “Large neighborhood search”. In: *International Series in Operations Research and Management Science*. Ed. by M. Gendreau and J.-Y. Potvin. Vol. 272. Cham: Springer International Publishing, pp. 99–127. URL: https://doi.org/10.1007/978-3-319-91086-4_4.

Porto, A. F., C. A. Henao, H. López-Ospina, and E. R. González (2019). “Hybrid flexibility strategy on personnel scheduling: Retail case study”. In: *Computers and Industrial Engineering* 133.January, pp. 220–230. URL: https://doi.org/10.1016/j.cie.2019.04.049.

Prot, D., T. Lapègue, and O. Bellenguez-Morineau (2015). “A two-phase method for the shift design and personnel task scheduling problem with equity objective”. In: *International Journal of Production Research* 53.24, pp. 7286–7298. URL: https://hal.archives-ouvertes.fr/hal-01061904.

Purnomo, H. W. and J. F. Bard (2007). “Cyclic preference scheduling for nurses using branch and price”. In: *Naval Research Logistics* 54, pp. 200–220.

Rekik, M., J. F. Cordeau, and F. Soumis (2010). “Implicit shift scheduling with multiple breaks and work stretch duration restrictions”. In: *Journal of Scheduling* 13.1, pp. 49–75.

Rocha, M., J. F. Oliveira, and M. A. Carravilla (Apr. 2013). “Cyclic staff scheduling: Optimization models for some real-life problems”. In: *Journal of Scheduling* 16.2, pp. 231–242.

Rönnberg, E. and T. Larsson (2010). “Automating the self-scheduling process of nurses in Swedish healthcare: A pilot study”. In: *Health Care Management Science* 13.1, pp. 35–53.

Ropke, S. (2009). “Parallel large neighborhood search - a software framework”. In: *MIC 2009: The VIII Metaheuristics International Conference*.

Ropke, S. and D. Pisinger (2006). “An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows”. In: *Transportation Science* 40.4, pp. 455–472.

Ropke, S. and A. Santini (2019). “Parallel Adaptive Large Neighborhood Search”. In: *Manuscript in preparation*.

Santini, A. (2019). *Adaptive Large Neighbourhood Search*. URL: https://github.com/alberto-santini/adaptive-large-neighbourhood-search.

Santini, A., S. Ropke, and L. M. Hvattum (2018). “A comparison of acceptance criteria for the adaptive large neighbourhood search metaheuristic”. In: *Journal of Heuristics* 24.5, pp. 783–815. URL: https://doi.org/10.1007/s10732-018-9377-x.

Savelsbergh, M. W. P. (2001). “Branch and price: Integer programming wish column generationBranch and Price: Integer Programming with Column Generation”. In: *Encyclopedia of Optimization*. Ed. by C. A. Floudas and P. M. Pardalos. Boston, MA: Springer US, pp. 218–221. URL: https://doi.org/10.1007/0-306-48332-7_47.

Schrimpf, G., J. Schneider, H. Stamm-Wilbrandt, and G. Dueck (2000). “Record Breaking Optimization Results Using the Ruin and Recreate Principle”. In: *Journal of Computational Physics* 159.2, pp. 139–171.

Shaw, P. (1997). “A new local search algorithm providing high quality solutions to vehicle routing problems”. In: *APES Group, Dept of Computer Science, University of ...*, pp. 1–12.

Smet, P., S. Martin, D. Ouelhadj, E. Özcan, et al. (2012). "Investigation of fairness measures for nurse rostering". In: *PATAT 2012 - Proceedings of the 9th International Conference on the Practice and Theory of Automated Timetabling*. August, pp. 369–372. URL: http://eprints.port.ac.uk/6441/.

Smet, P. and G. Vanden (2016). "Large neighbourhood search for large-scale shift assignment problems with multiple tasks". In: *Proceedings of the 11th International Confenference on Practice and Theory of Automated Timetabling (PATAT 2016)*, pp. 339–352.

Stark, C. and J. Zimmermann (2005). "An Exact Branch-and-Price Algorithm for Workforce Scheduling". In: *Operations Research Proceedings 2004*. Ed. by H. Fleuren, D. den Hertog, and P. Kort. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 207–212.

Stølevik, M., T. E. Nordlander, A. Riise, and H. Frøyseth (2011). "A hybrid approach for solving real-world nurse rostering problems". In: *Lecture Notes in Computer Science* 6876.0314, pp. 85–99.

Thompson, G. M. (1995). "Improved Implicit Optimal Modeling of the Labor Shift Scheduling Problem". In: *Management Science* 41.4, pp. 595–607.

Tien, J. M. and A. Kamiyama (1982). "On Manpower Scheduling Algorithms". In: *SIAM Review* 24.3, pp. 275–287.

Van Den Bergh, J., J. Beliën, P. De Bruecker, E. Demeulemeester, et al. (2013). "Personnel scheduling: A literature review". In: *European Journal of Operational Research* 226.3, pp. 367–385. URL: http://dx.doi.org/10.1016/j.ejor.2012.11.029.

Vissers, J. M., J. W. M. Bertrand, and G. De Vries (2001). "A framework for production control in health care organizations". In: *Production Planning and Control* 12.6, pp. 591–604.

Wright, P. D., K. M. Bretthauer, and M. J. Côté (2006). "Reexamining the nurse scheduling problem: Staffing ratios and nursing shortages". In: *Decision Sciences* 37.1, pp. 39–70.

Wu, T. H., J. Y. Yeh, and Y. M. Lee (2015). "A particle swarm optimization approach with refinement procedure for nurse rostering problem". In: *Computers and Operations Research* 54, pp. 52–63. URL: http://dx.doi.org/10.1016/j.cor.2014.08.016.

Zijm, W. (2000). "Towards intelligent manufacturing planning and control systems". In: *OR Spektrum* 22, p. 313.

Appendix A

Model Formulations

Appendix A presents the mathematical models discussed in this master thesis. In Appendix A.1 the Shift Reduction extension is presented, while Appendix A.2 presents the mathematical model described in Chapter 6. Appendix A.3 presents the mathematical model that was developed in the preliminary work.

A.1 Shift Reduction Extension

Sets

$\mathcal{T}$: Set of time periods with demand in planning horizon

$\mathcal{S}$: Set of shifts

$\mathcal{S}_t^T$: Set of shifts overlapping time t

$\mathcal{S}^L$: Set of shifts with duration longer than desired duration

$\mathcal{S}^S$: Set of shifts with duration shorter than desired duration

$\mathcal{C}$: Set of competencies

Parameters

$\underline{D}_{ct}$: Minimum coverage of demand for competency type c at time t

D_{ct} : Ideal coverage of demand for competency type c at time t

$\overline{D}_{ct}$: Maximum coverage of demand for competency type c at time t

$\overline{V}$: Maximal desired shift duration

$\underline{V}$: Minimal desired shift duration

V_s : Duration of shift s

M : Big M for mapping use of shift

W^{D+} : Weight for positive deviation between actual and ideal demand coverage

W^{D-} : Weight for negative deviation between actual and ideal demand coverage

W^{L} : Weight for use of shifts with durations exceeding desired limit

W^{S} : Weight for use of shifts with durations shorter than desired limit

Variables

$$y_s = \begin{cases} 1, & \text{if shift } s \text{ is used} \\ 0, & \text{otherwise} \end{cases}$$

x_s = Number of times shift s is used

μ_t = difference between covered demand and minimum demand at time t

$\delta_t^+,\ \delta_t^-$ = excess/deficit on coverage of demand at time t relative to ideal demand

Objective Function

$$\min z = \sum_{s \in \mathcal{S}} y_s \tag{A.1a}$$

$$+ W^D \cdot \sum_{t \in \mathcal{T}} (\delta_t^+ + \delta_t^-) \tag{A.1b}$$

$$+ W^L \cdot \sum_{s \in \mathcal{S}^L} (V_s - \overline{V}) \cdot y_s + W^S \cdot \sum_{s \in \mathcal{S}^S} (\underline{V} - V_s) \cdot y_s \tag{A.1c}$$

Constraints

$$\sum_{s \in \mathcal{S}_t^T} x_s = \sum_{c \in \mathcal{C}} \underline{D}_{ct} + \mu_t \qquad t \in \mathcal{T} \tag{A.2}$$

$$\mu_t \leq \sum_{c \in \mathcal{C}} (\overline{D}_{ct} - \underline{D}_{ct}) \qquad t \in \mathcal{T} \tag{A.3}$$

$$\mu_t + \sum_{c \in \mathcal{C}} (\underline{D}_{ct} - D_{ct}) = \delta_t^+ - \delta_t^- \qquad t \in \mathcal{T} \tag{A.4}$$

$$x_s \leq M \cdot y_s \qquad s \in \mathcal{S} \tag{A.5}$$

Constraints on Variable

$$y_s \in \{0, 1\} \qquad s \in \mathcal{S} \tag{A.6}$$

$$x_s \geq \mathbb{Z}^+ \qquad s \in \mathcal{S} \tag{A.7}$$

$$\mu_t \geq \mathbb{Z}^+ \qquad t \in \mathcal{T} \tag{A.8}$$

$$\delta_t^+,\ \delta_t^- \geq \mathbb{Z}^+ \qquad t \in \mathcal{T} \tag{A.9}$$

A.2 Mathematical Model

Sets

$\mathcal{C}$: Set of competencies

$\mathcal{E}$: Set of employees

$\mathcal{E}_c^C$: Set of employees with competency c

$\mathcal{I}$: Set of days in planning horizon

$\mathcal{I}^{SAT}$: Set of Saturdays, indicating start of weekend

$\mathcal{J}$: Set of weeks in planning horizon

$\mathcal{T}$: Set of time periods with demand in the planning horizon

$\mathcal{T}_r^R$: Set of time periods with demand in weekly off-shift r

$\mathcal{S}$: Set of work shifts

$\mathcal{S}_i^I$: Set of possible work shifts at day i

$\mathcal{S}_t^T$: Set of work shifts overlapping time t

$\mathcal{S}_e^{INV}$: Set of invalid work shifts for employee e

$\mathcal{S}_{es}^{DRC}$: Set of work shifts that violates daily rest for employee e if shift s is worked

$\mathcal{S}_{es}^{DRS}$: Set of work shifts that violates daily rest for employee e if two or more of them are worked in addition to shift s

$\mathcal{R}$: Set of weekly off-shifts

$\mathcal{R}_j^J$: Set of weekly off-shifts in week j

Parameters

$\underline{D}_{ct}$: Minimum coverage of demand for competency type c at time t

D_{ct} : Ideal coverage of demand for competency type c at time t

$\overline{D}_{ct}$: Maximum allowed coverage of demand for competency type c at time t

V_r^R : Duration of weekly off-shift r

$\overline{V}^R$: Maximum rewarded weekly off-shift duration

B_e : Amount of weekly contracted time periods for employee e

L^{CD} : Desired limit of consecutive working days

P_{et} : Preference for working the time period t for employee e. Negative value indicating preference for not working, positive value indicating preference for work and 0 indicating indifference

W^{D+} : Weight for positive deviation between actual and ideal demand coverage

W^{D-} : Weight for negative deviation between actual and ideal demand coverage

W^{F} : Weight for fairness score of least favored employee

W^{R} : Weight for maximum continuous weekly rest

W^{B} : Weight for deviation between worked and contracted hours

W^{PW} : Weight for partial weekends

W^{IW} : Weight for isolated working days

W^{IO} : Weight for isolated off-days

W^{CD} : Weight for consecutive days

W^{P} : Weight for granting preferences

A^{D} : Adjusting factor for weighting of demand deviation

A^{F} : Adjusting factor for weighting of fairness score of least favoured employee

A_e^{B} : Adjusting factor for weighting of deviation in contracted hours for employee e

A^{P} : Adjusting factor for weighting of preferences

A.2.1 Variables

$$x_{es} = \begin{cases} 1, & \text{if employee } e \text{ works shift } s \\ 0, & \text{otherwise} \end{cases}$$

$$y_{cet} = \begin{cases} 1, & \text{if employee } e \text{ covers one unit of demand for competence } c \text{ at time } t \\ 0, & \text{otherwise} \end{cases}$$

$$w_{er} = \begin{cases} 1, & \text{if employee } e \text{ takes the weekly off-shift } r \\ 0, & \text{otherwise} \end{cases}$$

f_e = variable storing the fairness score of employee e

g = variable storing the lowest fairness score among the employees

γ_{ei} = $\begin{cases} 1, & \text{if employee } e \text{ works day } i \\ 0, & \text{otherwise} \end{cases}$

$\rho_{ei}^{SAT}, \rho_{ei}^{SUN}$ = $\begin{cases} 1, & \text{if employee } e \text{ works Saturday/Sunday and not Sunday/Saturday on day } i \\ 0, & \text{otherwise} \end{cases}$

σ_{ei} = $\begin{cases} 1, & \text{if employee } e \text{ has an isolated work day on day } i \\ 0, & \text{otherwise} \end{cases}$

ϕ_{ei} = $\begin{cases} 1, & \text{if employee } e \text{ has an isolated off-day on day } i \\ 0, & \text{otherwise} \end{cases}$

π_{ei} = $\begin{cases} 1, & \text{if employee } e \text{ on day } i \text{ starts a sequence of consecutive working days exceeding desired limit} \\ 0, & \text{otherwise} \end{cases}$

$\delta_{ct}^{+},\ \delta_{ct}^{-}$ = excess/deficit on coverage of demand for competency c at time t according to the ideal demand

λ_e = deviation in worked and contracted time periods for employee e

μ_{ct} = difference between covered demand and minimum demand for competency type c and shift type s at time t

Objective Function

$$\max z = \sum_{e \in \mathcal{E}} f_e \tag{A.10a}$$

$$+ W^F \cdot A^F \cdot g \tag{A.10b}$$

$$- A^D \cdot \sum_{c \in \mathcal{C}} \sum_{t \in \mathcal{T}} (W^{D+} \cdot \delta_{ct}^{+} \ + \ W^{D-} \cdot \delta_{ct}^{-}) \tag{A.10c}$$

Constraints

Demand Covering

$$\sum_{e \in \mathcal{E}_c^C} y_{cet} = \underline{D}_{ct} + \mu_{ct} \qquad c \in \mathcal{C},\ t \in \mathcal{T} \tag{A.11}$$

$$\mu_{ct} \leq \overline{D}_{ct} - \underline{D}_{ct} \qquad c \in \mathcal{C},\ t \in \mathcal{T} \tag{A.12}$$

$$\mu_{ct} + \underline{D}_{ct} - D_{ct} = \delta^{+}_{ct} - \delta^{-}_{ct} \qquad c \in \mathcal{C},\ t \in \mathcal{T} \tag{A.13}$$

Work Allocation

$$\sum_{s \in \mathcal{S}^{I}_{i}} x_{es} = \gamma_{ei} \qquad e \in \mathcal{E},\ i \in \mathcal{I} \tag{A.14}$$

$$\sum_{s \in \mathcal{S}^{T}_{t}} x_{es} = \sum_{c \in \mathcal{C}} y_{cet} \qquad e \in \mathcal{E},\ t \in \mathcal{T} \tag{A.15}$$

$$\sum_{c \in \mathcal{C}} y_{cet} \leq 1 \qquad e \in \mathcal{E},\ t \in \mathcal{T} \tag{A.16}$$

Rest

$$2 \cdot x_{es} + \sum_{s' \in \mathcal{S}^{DRC}_{es}} x_{es'} \leq 2 \qquad e \in \mathcal{E},\ s \in \mathcal{S} \mid \mathcal{S}^{DRC}_{es} \neq \emptyset \tag{A.17}$$

$$x_{es} + \sum_{s' \in \mathcal{S}^{DRS}_{es}} x_{es'} \leq 2 \qquad e \in \mathcal{E},\ s \in \mathcal{S} \mid \mathcal{S}^{DRS}_{es} \neq \emptyset \tag{A.18}$$

$$\sum_{r \in \mathcal{R}^{J}_{j}} w_{er} = 1 \qquad e \in \mathcal{E},\ j \in \mathcal{J} \tag{A.19}$$

$$|\mathcal{T}^{R}_{r}| \cdot w_{er} \leq \sum_{t \in \mathcal{T}^{R}_{r}} 1 - \sum_{c \in \mathcal{C}} y_{cet} \qquad e \in \mathcal{E},\ r \in \mathcal{R} \tag{A.20}$$

Contracted hours

$$\sum_{c \in \mathcal{C}} \sum_{t \in \mathcal{T}} y_{cet} + \lambda_{e} = |\mathcal{J}| \cdot B_{e} \qquad e \in \mathcal{E}, \tag{A.21}$$

Partial Weekends

$$\gamma_{ei} - \gamma_{e(i+1)} = \rho^{SAT}_{ei} - \rho^{SUN}_{e(i+1)} \qquad e \in \mathcal{E},\ i \in \mathcal{I}^{SAT} \tag{A.22}$$

Isolated Days

$$-\gamma_{e(i-1)} + \gamma_{ei} - \gamma_{e(i+1)} \leq \sigma_{ei} \qquad e \in \mathcal{E},\ i \in \{2, ..., |\mathcal{I}| - 1\} \tag{A.23}$$

$$\gamma_{e(i-1)} - \gamma_{ei} + \gamma_{e(i+1)} - 1 \leq \phi_{ei} \qquad e \in \mathcal{E},\ i \in \{2, ..., |\mathcal{I}| - 1\} \tag{A.24}$$

Consecutive Working Days

$$\sum_{i'=i}^{i'+L^{CD}} \gamma_{ei'} - L^{CD} \leq \pi_{ei} \qquad e \in \mathcal{E},\ i \in \{1, ..., |\mathcal{I}| - L^{CD}\} \tag{A.25}$$

Fairness

$$\begin{aligned}
f_e = \quad & W^R \cdot \sum_{r \in \mathcal{R}} \min(V^R_r, \overline{V}^R) \cdot w_{er} && && \text{(A.26a)}\\
& - W^B \cdot A^B_e \cdot \lambda_e && && \text{(A.26b)}\\
& - W^{PW} \cdot \sum_{i \in \mathcal{I}^{SAT}} (\rho^{SAT}_{ei} + \rho^{SUN}_{e(i+1)}) && && \text{(A.26c)}\\
& - W^{IW} \cdot \sum_{i \in \mathcal{I}} \sigma_{ei} && && \text{(A.26d)}\\
& - W^{IO} \cdot \sum_{i \in \mathcal{I}} \phi_{ei} && && \text{(A.26e)}\\
& - W^{CD} \cdot \sum_{i \in \mathcal{I}} \pi_{ei} && && \text{(A.26f)}\\
& + W^P \cdot \sum_{t \in \mathcal{T}} P_{et} \cdot \sum_{c \in C} y_{cet} && e \in \mathcal{E} && \text{(A.26g)}
\end{aligned}$$

Constraints on Variables

$$x_{es} \in \{0, 1\} \qquad e \in \mathcal{E},\ s \in \mathcal{S} \setminus \mathcal{S}^{INV}_e \tag{A.27}$$

$$y_{cet} \in \{0,1\} \qquad c \in \mathcal{C},\ e \in \mathcal{E},\ t \in \mathcal{T} \tag{A.28}$$

$$w_{er} \in \{0,1\} \qquad e \in \mathcal{E},\ r \in \mathcal{R} \tag{A.29}$$

$$\gamma_{ei} \in \{0,1\} \qquad e \in \mathcal{E},\ i \in \mathcal{I} \tag{A.30}$$

$$\rho_{ei}^{SAT}, \rho_{e(i+1)}^{SUN} \in \{0,1\} \qquad e \in \mathcal{E},\ i \in \mathcal{I}^{SAT} \tag{A.31}$$

$$\sigma_{ei} \in \{0,1\} \qquad e \in \mathcal{E},\ i \in \mathcal{I} \tag{A.32}$$

$$\phi_{ei} \in \{0,1\} \qquad e \in \mathcal{E},\ i \in \mathcal{I} \tag{A.33}$$

$$\pi_{ei} \in \{0,1\} \qquad e \in \mathcal{E},\ i \in \mathcal{I} \tag{A.34}$$

$$\delta_{ct}^{+},\ \delta_{ct}^{-} \geq \mathbb{Z}^{+} \qquad c \in \mathcal{C},\ t \in \mathcal{T} \tag{A.35}$$

$$\lambda_e \geq \mathbb{Z}^{+} \qquad e \in \mathcal{E} \tag{A.36}$$

$$\mu_{ct} \geq \mathbb{Z}^{+} \qquad c \in \mathcal{C},\ s \in \mathcal{S},\ t \in \mathcal{T} \tag{A.37}$$

$$f_e = free \qquad e \in \mathcal{E} \tag{A.38}$$

$$g = free \tag{A.39}$$

A.3 Preliminary Work: Mixed Integer Programming Model with Implicit Shift Design

Sets

$\mathcal{E}$: Set of employees, $e \in \{1, ..., |\mathcal{E}|\}$

$\mathcal{E}_c^C$: Set of employees with competency c, $e \in \{1, ..., |\mathcal{E}_c^C|\}$

$\mathcal{I}$: Set of universal days in the planning period, $i \in \{1, ..., |\mathcal{I}|\}$

$\mathcal{I}^{Sat}$: Set of Saturdays, indicating start of weekend, $i \in \{\underline{\mathcal{I}}^{Sat}, ..., \overline{\mathcal{I}}^{Sat}\}$

$\mathcal{J}$: Set of all universal weeks in the planning period, $j \in \{1, ..., |\mathcal{J}|\}$

$\mathcal{T}$: Set of time periods in the planning horizon, $t \in \{1, ..., |\mathcal{T}|\}$

$\mathcal{T}_i^I$: Set of time periods in universal day i, $t \in \{\underline{\mathcal{T}}_i^I, ..., \overline{\mathcal{T}}_i^I\}$

$\mathcal{S}$: Set of shift types, $s \in \{\text{Work, On duty, On call, Off}\}$

$\mathcal{S}^W$: Set of work shifts, $s \in \mathcal{S} \setminus \{\text{Off}\}$

$\mathcal{C}$: Set of competencies, $c \in \{1, ..., |\mathcal{C}|\}$

$\mathcal{V}^W$: Set of possible work shift durations, $v \in \{\underline{\mathcal{V}}^W, ..., \overline{\mathcal{V}}^W\}$

$\mathcal{V}^I$: Set of possible daily off shift durations, $v \in \{\underline{\mathcal{V}}^I, ..., \overline{\mathcal{V}}^I\}$

$\mathcal{V}^J$: Set of possible weekly off shift durations, $v \in \{\underline{\mathcal{V}}^J, ..., \overline{\mathcal{V}}^J\}$

Parameters

$\underline{D}_{cst}$: Minimum coverage of demand for competency type c and shift type s at time t

D_{cst} : Ideal coverage of demand for competency type c and shift type s at time t

$\overline{D}_{cst}$: Maximum allowed coverage of demand for competency type c and shift type s at time t

N : Time resolution

L^{CD} : Desired limit of consecutive universal working days

B_e : Amount of weekly contracted time periods for employee e

R_e^I : The required continuous daily rest in time periods for employee e

R_e^J : The required continuous weekly rest in time periods for employee e

K_e^I : The offset from universal day in time periods for employee e

K_e^J : The offset from universal weeks in time periods for employee e

H^J : Number of time periods in a week

P_{et} : Preference for working the time period t for employee e. Negative value indicating preference for not working, positive value indicating preference for work and 0 indicating indifference

W^{D+} : Weight for positive deviation between actual and ideal demand coverage

W^{D-} : Weight for negative deviation between actual and ideal demand coverage

W^{R} : Weight for maximum continuous weekly rest

W^{B} : Weight for deviation between worked and contracted hours

W^{P} : Weight for working a undesired time

W^{F} : Weight for fairness score of least favored employee

W^{PW} : Weight for working partial weekends

W^{IW} : Weight for isolated working day

W^{IO} : Weight for isolated off day

W^{CD} : Weight for working more than desired amount of consecutive days

A^{D} : Adjusting factor for weighting of demand deviation

A^{F} : Adjusting factor for weighting of fairness score of least favoured employee

A^{B}_{e} : Adjusting factor for weighting of deviation in contracted hours for employee e

A^{P} : Adjusting factor for weighting of preferences

Variables

$$x_{estv} = \begin{cases} 1, & \text{if employee } e \text{ starts a shift type } s \text{ at time } t \text{ with duration } v \\ 0, & \text{otherwise} \end{cases}$$

$$y_{cet} = \begin{cases} 1, & \text{if employee } e \text{ covers one unit of demand for competence type } c \text{ at time } t \\ 0, & \text{otherwise} \end{cases}$$

f_e = variable storing the fairness score of employee e

g = variable storing the lowest fairness score among the employees

γ_{ei} = $\begin{cases} 1, & \text{if employee } e \text{ works on universal day } i \\ 0, & \text{otherwise} \end{cases}$

$\rho_{ei}^{Sat}, \rho_{ei}^{Sun}$ = $\begin{cases} 1, & \text{if employee } e \text{ works Saturday/Sunday and not Sunday/Saturday on universal day } i \\ 0, & \text{otherwise} \end{cases}$

σ_{ei} = $\begin{cases} 1, & \text{if employee } e \text{ has an isolated work day on universal day } i \\ 0, & \text{otherwise} \end{cases}$

ϕ_{ei} = $\begin{cases} 1, & \text{if employee } e \text{ has an isolated off-day on universal day } i \\ 0, & \text{otherwise} \end{cases}$

π_{ei} = $\begin{cases} 1, & \text{if employee } e \text{ on universal day } i \text{ starts a sequence of consecutive working days exceeding desired limit} \\ 0, & \text{otherwise} \end{cases}$

λ_e = difference in worked and contracted hours for employee e

μ_{cst} = difference between covered demand and minimum demand for competency type c and shift type s at time t

$\delta_{cst}^{+}, \delta_{cst}^{-}$ = Excess/deficit on coverage of demand for competency c and shift type s at timer t according to the ideal demand

Objective Function

$$\max z = \sum_{e \in \mathcal{E}} f_e + W^F \cdot A^F \cdot g - A^D \cdot \sum_{c \in \mathcal{C}} \sum_{t \in \mathcal{T}} (W^{D+} \cdot \delta_{ct}^{+} \; + \; W^{D-} \cdot \delta_{ct}^{-}) \tag{A.40}$$

Constraints

Demand Covering

$$\sum_{e \in \mathcal{E}_c} y_{cet} = \underline{D}_{cst} + \mu_{cst} \qquad c \in \mathcal{C},\ s \in \mathcal{S}^W,\ t \in \mathcal{T} \tag{A.41}$$

$$\mu_{cst} \leq \overline{D}_{cst} - \underline{D}_{cst} \qquad c \in \mathcal{C},\ s \in \mathcal{S}^W,\ t \in \mathcal{T} \tag{A.42}$$

$$\mu_{cst} + \underline{D}_{cst} - D_{cst} = \delta^{+}_{cst} - \delta^{-}_{cst} \qquad c \in \mathcal{C},\ s \in \mathcal{S}^W,\ t \in \mathcal{T} \tag{A.43}$$

Work Allocation

$$\sum_{s \in \mathcal{S}^W} \sum_{v \in \mathcal{V}^I} \sum_{t'=t-v+N}^{t} x_{est'v} = \sum_{c \in \mathcal{C}} y_{cet} \qquad e \in \mathcal{E},\ t \in \mathcal{T} \tag{A.44}$$

$$\sum_{c \in \mathcal{C}} y_{cet} \leq 1 \qquad e \in \mathcal{E},\ t \in \mathcal{T} \tag{A.45}$$

$$\sum_{s \in S^W} \sum_{t \in \mathcal{T}^I_i} \sum_{v \in \mathcal{V}^W} x_{estv} \leq 1 \qquad e \in \mathcal{E},\ i \in \mathcal{I} \tag{A.46}$$

$$v \cdot x_{estv} \leq N \cdot \sum_{t'=t}^{t+v-N} (1 - \sum_{c \in \mathcal{C}} y_{cet'}) \qquad e \in \mathcal{E},\ s = \text{Off},\ t \in \mathcal{T},\ v \in \mathcal{V}^I \cup \mathcal{V}^J \mid v \geq R^I_e \tag{A.47}$$

Rest

$$\sum_{v=R^I_e}^{|\mathcal{V}^I|} \sum_{t=1}^{|T^I_i|-v} x_{es(t+K^I_e)v} = 1 \qquad e \in \mathcal{E},\ i \in \mathcal{I}, s = \text{Off} \tag{A.48}$$

$$\sum_{v=R^J_e}^{|\mathcal{V}^J|} \sum_{t=K^J_e+(j-1)\cdot H^J}^{K^J_e+j\cdot H^J-v} x_{estv} = 1 \qquad e \in \mathcal{E},\ j \in \mathcal{J},\ s = \text{Off} \tag{A.49}$$

Contracted hours

$$N \cdot \sum_{c \in \mathcal{C}} \sum_{t \in \mathcal{T}} y_{cet} + \lambda_e = |\mathcal{J}| \cdot B_e \qquad e \in \mathcal{E}, \tag{A.50}$$

Helper Variable

$$N \cdot \sum_{c \in \mathcal{C}} \sum_{t \in \mathcal{T}_i^I} y_{cet} \leq M^I \cdot \gamma_{ei} \qquad e \in \mathcal{E},\ i \in \mathcal{I} \tag{A.51}$$

$$N \cdot \sum_{c \in \mathcal{C}} \sum_{t \in \mathcal{T}_i^I} y_{cet} \geq M^I \cdot (\gamma_{ei} - 1) + N \qquad e \in \mathcal{E},\ i \in \mathcal{I} \tag{A.52}$$

Partial Weekends

$$\gamma_{ei} - \gamma_{e(i+1)} = \rho_{ei}^{SAT} - \rho_{e(i+1)}^{SUN} \qquad e \in \mathcal{E},\ i \in \mathcal{I}^{Sat} \tag{A.53}$$

Isolated Days

$$-\gamma_{ei} + \gamma_{e(i+1)} - \gamma_{e(i+2)} \leq \sigma_{e(i+1)} \qquad e \in \mathcal{E},\ i \in \{1, ..., |\mathcal{I}| - 2\} \tag{A.54}$$

$$\gamma_{ei} - \gamma_{e(i+1)} + \gamma_{e(i+2)} - 1 \leq \phi_{e(i+1)} \qquad e \in \mathcal{E},\ i \in \{1, ..., |\mathcal{I}| - 2\} \tag{A.55}$$

Consecutive Working Days

$$\sum_{i'=i}^{i+L^{CD}} w_{ei'} - L^{CD} \leq M^{CD} \cdot \pi_{ei} - 1 \qquad e \in \mathcal{E},\ i \in \{1, ..., |\mathcal{I}| - L^{CD}\} \tag{A.56}$$

Fairness

$$
\begin{aligned}
f_e = \quad & + W^R \cdot \sum_{t \in \mathcal{T}} \sum_{v \in \mathcal{V}^J} v \cdot x_{e(s=\text{off})tv} && \text{(A.57a)}\\
& - W^B \cdot \lambda_e && \text{(A.57b)}\\
& - W^{PW} \cdot \sum_{j \in \mathcal{J}} (\rho_{ej}^{sat} + \rho_{ej}^{sun}) && \text{(A.57c)}\\
& - W^{IW} \cdot \sum_{i \in \mathcal{I}} \sigma_{ei} && \text{(A.57d)}\\
& - W^{IO} \cdot \sum_{i \in \mathcal{I}} \phi_{ei} && \text{(A.57e)}\\
& - W^{CD} \cdot \sum_{i \in \mathcal{I}} \pi_{ei} && \text{(A.57f)}\\
& + W^P \cdot \sum_{t \in \mathcal{T}} P_{et} \sum_{c \in C} y_{cet} \qquad e \in \mathcal{E} && \text{(A.57g)}\\
& && \text{(A.57h)}
\end{aligned}
$$

$$g \leq f_e \qquad e \in \mathcal{E} \tag{A.58}$$

Constraints on Variables

$$x_{estv} \in \{0,1\} \qquad e \in \mathcal{E},\ s \in \mathcal{S},\ t \in \mathcal{T},\ v \in \mathcal{V} \tag{A.59}$$

$$y_{cet} \in \{0,1\} \qquad c \in \mathcal{C},\ e \in \mathcal{E},\ t \in \mathcal{T} \tag{A.60}$$

$$f_e = free \qquad e \in \mathcal{E} \tag{A.61}$$

$$g = free \tag{A.62}$$

$$\gamma_{ei} \in \{0,1\} \qquad e \in \mathcal{E},\ i \in \mathcal{I} \tag{A.63}$$

$$\rho_{ei}^{Sat},\ \rho_{ei}^{Sun} \in \{0,1\} \qquad e \in \mathcal{E},\ i \in \mathcal{I} \tag{A.64}$$

$$\sigma_{ei} \ \in \{0,1\} \qquad e \in \mathcal{E},\ i \in \mathcal{I} \tag{A.65}$$

$$\phi_{ei} \in \{0,1\} \qquad e \in \mathcal{E},\ i \in \mathcal{I} \tag{A.66}$$

$$\pi_{ei} \in \{0,1\} \qquad e \in \mathcal{E},\ i \in \mathcal{I} \tag{A.67}$$

$$\delta_t^+,\ \delta_t^- \geq \mathbb{N}_0 \qquad t \in \mathcal{T} \tag{A.68}$$

$$\lambda_e \geq 0 \qquad e \in \mathcal{E} \tag{A.69}$$

$$\mu_{cst} \geq \mathbb{N}_0 \qquad c \in \mathcal{C},\ s \in \mathcal{S},\ t \in \mathcal{T} \tag{A.70}$$

A.4 Construction Model

Sets

$\mathcal{C}$	:	Set of competencies
$\mathcal{E}$	:	Set of employees
$\mathcal{E}_c^C$	:	Set of employees with competency c
$\mathcal{I}$	:	Set of days in planning horizon
$\mathcal{J}$	:	Set of weeks in planning horizon
$\mathcal{T}$	:	Set of time periods with demand in the planning horizon
$\mathcal{T}_r^R$	:	Set of time periods with demand in weekly off-shift r
$\mathcal{S}$	:	Set of work shifts
$\mathcal{S}_i^I$	:	Set of possible work shifts at day i
$\mathcal{S}_t^T$	:	Set of work shifts overlapping time t
$\mathcal{S}_e^{INV}$	:	Set of invalid work shifts for employee e
$\mathcal{S}_{es}^{DRC}$	:	Set of work shifts that violates daily rest for employee e if shift s is worked
$\mathcal{S}_{es}^{DRS}$	:	Set of work shifts that violates daily rest for employee e if two or more of them are worked in addition to shift s
$\mathcal{R}$	:	Set of weekly off-shifts
$\mathcal{R}_j^J$	:	Set of weekly off-shifts in week j

Parameters

$\underline{D}_{ct}$	:	Minimum coverage of demand for competency type c at time t
D_{ct}	:	Ideal coverage of demand for competency type c at time t
$\overline{D}_{ct}$	:	Maximum allowed coverage of demand for competency type c at time t
V_r^R	:	Duration of weekly off-shift r
$\overline{V}^R$	:	Maximum rewarded weekly off-shift duration
B_e	:	Amount of weekly contracted time periods for employee e

A.4.1 Variables

x_{es} = $\begin{cases} 1, & \text{if employee } e \text{ works shift } s \\ 0, & \text{otherwise} \end{cases}$

y_{cet} = $\begin{cases} 1, & \text{if employee } e \text{ covers one unit of demand for competence } c \text{ at time } t \\ 0, & \text{otherwise} \end{cases}$

w_{er} = $\begin{cases} 1, & \text{if employee } e \text{ takes the weekly off-shift } r \\ 0, & \text{otherwise} \end{cases}$

f_e = variable storing the fairness score of employee e

g = variable storing the lowest fairness score among the employees

γ_{ei} = $\begin{cases} 1, & \text{if employee } e \text{ works day } i \\ 0, & \text{otherwise} \end{cases}$

$\delta_{ct}^{+},\ \delta_{ct}^{-}$ = excess/deficit on coverage of demand for competency c at time t according to the ideal demand

λ_e = deviation in worked and contracted time periods for employee e

μ_{ct} = difference between covered demand and minimum demand for competency type c and shift type s at time t

Objective Function

$$\min \quad f'(x) = \sum_{c \in \mathcal{C}} \sum_{e \in \mathcal{E}} \sum_{t \in \mathcal{T}} y_{cet} \tag{A.71}$$

Constraints

Demand Covering

$$\sum_{e \in \mathcal{E}_c^{\mathcal{C}}} y_{cet} = \underline{D}_{ct} + \mu_{ct} \qquad c \in \mathcal{C},\ t \in \mathcal{T} \tag{A.72}$$

$$\mu_{ct} \leq \overline{D}_{ct} - \underline{D}_{ct} \qquad c \in \mathcal{C},\ t \in \mathcal{T} \tag{A.73}$$

$$\mu_{ct} + \underline{D}_{ct} - D_{ct} = \delta_{ct}^{+} - \ \delta_{ct}^{-} \qquad c \in \mathcal{C},\ t \in \mathcal{T} \tag{A.74}$$

Work Allocation

$$\sum_{s \in \mathcal{S}_i^I} x_{es} = \gamma_{ei} \qquad e \in \mathcal{E},\ i \in \mathcal{I} \tag{A.75}$$

$$\sum_{s \in \mathcal{S}_t^T} x_{es} = \sum_{c \in \mathcal{C}} y_{cet} \qquad e \in \mathcal{E},\ t \in \mathcal{T} \tag{A.76}$$

$$\sum_{c \in \mathcal{C}} y_{cet} \leq 1 \qquad e \in \mathcal{E},\ t \in \mathcal{T} \tag{A.77}$$

Rest

$$2 \cdot x_{es} + \sum_{s' \in \mathcal{S}_{es}^{DRC}} x_{es'} \leq 2 \qquad e \in \mathcal{E},\ s \in \mathcal{S} \mid \mathcal{S}_{es}^{DRC} \neq \emptyset \tag{A.78}$$

$$x_{es} + \sum_{s' \in \mathcal{S}_{es}^{DRS}} x_{es'} \leq 2 \qquad e \in \mathcal{E},\ s \in \mathcal{S} \mid \mathcal{S}_{es}^{DRS} \neq \emptyset \tag{A.79}$$

$$\sum_{r \in \mathcal{R}_j^J} w_{er} = 1 \qquad e \in \mathcal{E},\ j \in \mathcal{J} \tag{A.80}$$

$$|\mathcal{T}_r^R| \cdot w_{er} \leq \sum_{t \in \mathcal{T}_r^R} 1 - \sum_{c \in \mathcal{C}} y_{cet} \qquad e \in \mathcal{E},\ r \in \mathcal{R} \tag{A.81}$$

Contracted hours

$$\sum_{c \in \mathcal{C}} \sum_{t \in \mathcal{T}} y_{cet} + \lambda_e = |\mathcal{J}| \cdot B_e \qquad e \in \mathcal{E}, \tag{A.82}$$

Constraints on Variables

$$x_{es} \in \{0, 1\} \qquad e \in \mathcal{E},\ s \in \mathcal{S} \setminus \mathcal{S}_e^{INV} \tag{A.83}$$

$$y_{cet} \in \{0, 1\} \qquad c \in \mathcal{C},\ e \in \mathcal{E},\ t \in \mathcal{T} \tag{A.84}$$

$$w_{er} \in \{0, 1\} \qquad e \in \mathcal{E},\ r \in \mathcal{R} \tag{A.85}$$

$$\gamma_{ei} \in \{0, 1\} \qquad e \in \mathcal{E},\ i \in \mathcal{I} \tag{A.86}$$

$$\delta_{ct}^{+},\ \delta_{ct}^{-} \geq \mathbb{Z}^{+} \qquad c \in \mathcal{C},\ t \in \mathcal{T} \tag{A.87}$$

$$\lambda_{e} \geq \mathbb{Z}^{+} \qquad e \in \mathcal{E} \tag{A.88}$$

$$\mu_{ct} \geq \mathbb{Z}^{+} \qquad c \in \mathcal{C},\ s \in \mathcal{S},\ t \in \mathcal{T} \tag{A.89}$$

A.5 MIP Operators

A.5.1 Pure MIP Operator

Sets

$\mathcal{C}$: Set of competencies

$\mathcal{E}$: Set of employees

$\mathcal{E}_c^C$: Set of employees with competency c

$\mathcal{I}$: Set of days in destroyed weeks

$\mathcal{I}^{SAT}$: Set of Saturdays in destroyed weeks, indicating start of weekend

$\mathcal{J}$: Set of destroyed weeks

$\mathcal{T}$: Set of time periods with demand in the destroyed weeks

$\mathcal{S}$: Set of work shifts in destroyed weeks

$\mathcal{S}_i^I$: Set of possible work shifts at day i

$\mathcal{S}_t^T$: Set of work shifts overlapping time t

$\mathcal{S}_e^{INV}$: Set of invalid work shifts for employee e

$\mathcal{S}_{es}^{DRC}$: Set of work shifts that violates daily rest for employee e if shift s is worked

$\mathcal{S}_{es}^{DRS}$: Set of work shifts that violates daily rest for employee e if two or more of them are worked in addition to shift

Parameters

$\underline{D}_{ct}$: Minimum coverage of demand for competency type c at time t

D_{ct} : Ideal coverage of demand for competency type c at time t

$\overline{D}_{ct}$: Maximum allowed coverage of demand for competency type c at time t

B_e : Amount of weekly contracted time periods for employee e

L^{CD} : Desired limit of consecutive working days

P_{et} : Preference for working the time period t for employee e. Negative value indicating preference for not working, positive value indicating preference for work and 0 indicating indifference

W^P : Weight for working a undesired time

W^{PW} : Weight for working partial weekends

W^{IW} : Weight for isolated working day

W^{IO} : Weight for isolated off day

W^{CD} : Weight for working more than desired amount of consecutive days

W^{OCH} : Weight adjusting the impact of overuse of contracted hours.

A^{P} : Adjusting factor for weighting of preferences

Variables

x_{es} = $\begin{cases} 1, & \text{if employee } e \text{ works shift } s \\ 0, & \text{otherwise} \end{cases}$

y_{cet} = $\begin{cases} 1, & \text{if employee } e \text{ covers one unit of demand for competence } c \text{ at time } t \\ 0, & \text{otherwise} \end{cases}$

γ_{ei} = $\begin{cases} 1, & \text{if employee } e \text{ works day } i \\ 0, & \text{otherwise} \end{cases}$

$\rho_{ei}^{SAT}, \rho_{ei}^{SUN}$ = $\begin{cases} 1, & \text{if employee } e \text{ works Saturday/Sunday and not Sunday/Saturday on day } i \\ 0, & \text{otherwise} \end{cases}$

σ_{ei} = $\begin{cases} 1, & \text{if employee } e \text{ has an isolated work day on day } i \\ 0, & \text{otherwise} \end{cases}$

ϕ_{ei} = $\begin{cases} 1, & \text{if employee } e \text{ has an isolated off-day on day } i \\ 0, & \text{otherwise} \end{cases}$

π_{ei} = $\begin{cases} 1, & \text{if employee } e \text{ on day } i \text{ starts a sequence of consecutive working days exceeding desired limit} \\ 0, & \text{otherwise} \end{cases}$

λ_e^-, λ_e^+ = negative and positive difference in worked and contracted hours for employee e, updated with worked hours in non-destroyed weeks

μ_{ct} = difference between covered demand and minimum demand for competency type c at time t

$\delta_{ct}^+, \delta_{ct}^-$ = Excess/deficit on coverage of demand for competency c at time t according to the ideal demand

Objective Function

$$\begin{aligned}\max z = \quad & - W^{OCH} \cdot \sum_{e \in \mathcal{E}} \lambda_e^+ - W^{PW} \cdot \sum_{i \in \mathcal{I}^{SAT}} (\rho_{ei}^{SAT} + \rho_{e(i+1)}^{SUN}) \\ & - W^{IW} \cdot \sum_{i \in \mathcal{I}} \sigma_{ei} - W^{IO} \cdot \sum_{i \in \mathcal{I}} \phi_{ei} \\ & - W^{CD} \cdot \sum_{i \in \mathcal{I}} \pi_{ei} + A^P \cdot W^P \cdot \sum_{t \in \mathcal{T}} P_{et} \cdot \sum_{c \in C} y_{cet} \\ & - \sum_{c \in \mathcal{C}} \sum_{t \in \mathcal{T}} (W^{D+} \cdot \delta_{ct}^+ \; + \; W^{D-} \cdot \delta_{ct}^-)\end{aligned} \tag{A.90}$$

Constraints

$$\sum_{e \in \mathcal{E}_c^C} y_{cet} = \underline{D}_{ct} + \mu_{ct} \qquad c \in \mathcal{C},\ t \in \mathcal{T} \tag{A.91}$$

$$\mu_{ct} \leq \overline{D}_{ct} - \underline{D}_{ct} \qquad c \in \mathcal{C},\ t \in \mathcal{T} \tag{A.92}$$

$$\mu_{ct} + \underline{D}_{ct} - D_{ct} = \delta_{ct}^+ - \; \delta_{ct}^- \qquad c \in \mathcal{C},\ t \in \mathcal{T} \tag{A.93}$$

$$\sum_{s \in \mathcal{S}_i^I} x_{es} = \gamma_{ei} \qquad e \in \mathcal{E},\ i \in \mathcal{I} \tag{A.94}$$

$$\sum_{s \in \mathcal{S}_t^T} x_{es} = \sum_{c \in \mathcal{C}} y_{cet} \qquad e \in \mathcal{E},\ t \in \mathcal{T} \tag{A.95}$$

$$\sum_{c \in \mathcal{C}} y_{cet} \leq 1 \qquad e \in \mathcal{E},\ t \in \mathcal{T} \tag{A.96}$$

$$2 \cdot x_{es} + \sum_{s' \in \mathcal{S}_{es}^{DRC}} x_{es'} \leq 2 \qquad e \in \mathcal{E},\ s \in \mathcal{S} \mid \mathcal{S}_{es}^{DRC} \neq \emptyset \tag{A.97}$$

$$x_{es} + \sum_{s' \in \mathcal{S}_{es}^{DRS}} x_{es'} \leq 2 \qquad e \in \mathcal{E},\ s \in \mathcal{S} \mid \mathcal{S}_{es}^{DRS} \neq \emptyset \tag{A.98}$$

$$\sum_{c \in \mathcal{C}} \sum_{t \in \mathcal{T}} y_{cet} + \lambda_e^- - \lambda_e^+ = |\mathcal{J}| \cdot B_e \qquad e \in \mathcal{E}, \tag{A.99}$$

$$\gamma_{ei} - \gamma_{e(i+1)} = \rho_{ei}^{SAT} - \rho_{e(i+1)}^{SUN} \qquad e \in \mathcal{E},\ i \in \mathcal{I}^{SAT} \tag{A.100}$$

$$-\gamma_{e(i-1)} + \gamma_{ei} - \gamma_{e(i+1)} \leq \sigma_{ei} \qquad e \in \mathcal{E},\ i \in \{2, ..., |\mathcal{I}| - 1\} \tag{A.101}$$

$$\gamma_{e(i-1)} - \gamma_{ei} + \gamma_{e(i+1)} - 1 \leq \phi_{ei} \qquad e \in \mathcal{E},\ i \in \{2, ..., |\mathcal{I}| - 1\} \tag{A.102}$$

$$\sum_{i'=i}^{i'+L^{CD}} \gamma_{ei'} - L^{CD} \leq \pi_{ei} \qquad e \in \mathcal{E},\ i \in \{1, ..., |\mathcal{I}| - L^{CD}\} \tag{A.103}$$

Constraints on Variables

$$x_{es} \in \{0, 1\} \qquad e \in \mathcal{E},\ s \in \mathcal{S} \tag{A.104}$$

$$y_{cet} \in \{0, 1\} \qquad c \in \mathcal{C},\ e \in \mathcal{E},\ t \in \mathcal{T} \tag{A.105}$$

$$\gamma_{ei} \in \{0, 1\} \qquad e \in \mathcal{E},\ i \in \mathcal{I} \tag{A.106}$$

$$\rho_{ei}^{Sat},\ \rho_{ei}^{Sun} \in \{0, 1\} \qquad e \in \mathcal{E},\ i \in \mathcal{I} \tag{A.107}$$

$$\sigma_{ei} \in \{0, 1\} \qquad e \in \mathcal{E},\ i \in \mathcal{I} \tag{A.108}$$

$$\phi_{ei} \in \{0, 1\} \qquad e \in \mathcal{E},\ i \in \mathcal{I} \tag{A.109}$$

$$\pi_{ei} \in \{0, 1\} \qquad e \in \mathcal{E},\ i \in \mathcal{I} \tag{A.110}$$

$$\delta_t^+,\ \delta_t^- \geq \mathbb{N}_0 \qquad t \in \mathcal{T} \tag{A.111}$$

$$\lambda_e^-,\ \lambda_e^+ \geq 0 \qquad e \in \mathcal{E} \tag{A.112}$$

$$\mu_{ct} \geq \mathbb{N}_0 \qquad c \in \mathcal{C},\ s \in \mathcal{S},\ t \in \mathcal{T} \tag{A.113}$$

A.5.2 Hybrid MIP Operator

Sets

$\mathcal{C}$: Set of competencies

$\mathcal{I}_j$: Set of days in destroyed week j

$\mathcal{E}$: Set of employees

$\mathcal{I}^{SAT}$: Set of Saturdays in destroyed weeks, indicating start of weekend

$\mathcal{J}$: Set of destroyed weeks

$\mathcal{T}$: Set of time periods with demand in the destroyed weeks

$\mathcal{S}$: Set of work shifts in destroyed weeks

$\mathcal{S}_i^I$: Set of possible work shifts at day i

$\mathcal{S}_t^T$: Set of work shifts overlapping time t

Parameters

$\underline{D}_{ct}$: Minimum coverage of demand for competency type c at time t

D_{ct} : Ideal coverage of demand for competency type c at time t

$\overline{D}_{ct}$: Maximum allowed coverage of demand for competency type c at time t

Variables

$$x_s = \begin{cases} 1, & \text{if employee } e \text{ works shift } s \\ 0, & \text{otherwise} \end{cases}$$

$$y_{ct} = \begin{cases} 1, & \text{if employee } e \text{ covers one unit of demand for competence } c \text{ at time } t \\ 0, & \text{otherwise} \end{cases}$$

μ_{ct} = difference between covered demand and minimum demand for competency type c at time t

$\delta_{ct}^+, \delta_{ct}^-$ = Excess/deficit on coverage of demand for competency c at time t according to the ideal demand

Objective Function

$$Minimize \quad z = \sum_{t \in \mathcal{T}} \sum_{c \in \mathcal{C}} \delta_{ct}^+ + \delta_{ct}^- \tag{A.114}$$

Constraints

$$y_{ct} = \underline{D}_{ct} + \mu_{ct} \qquad c \in \mathcal{C},\ t \in \mathcal{T} \tag{A.115}$$

$$\mu_{ct} \leq \overline{D}_{ct} - \underline{D}_{ct} \qquad c \in \mathcal{C},\ t \in \mathcal{T} \tag{A.116}$$

$$\mu_{ct} + \underline{D}_{ct} - D_{ct} = \delta^{+}_{ct} - \ \delta^{-}_{ct} \qquad c \in \mathcal{C},\ t \in \mathcal{T} \tag{A.117}$$

$$\sum_{s \in \mathcal{S}^{T}_{t}} x_s = \sum_{c \in \mathcal{C}} y_{ct}, \qquad t \in \mathcal{T} \tag{A.118}$$

$$\sum_{s \in \mathcal{S}^{I}_{i}} x_s <= |\mathcal{E}| \qquad i \in \mathcal{I}_j \tag{A.119}$$

Constraints on Variables

$$x_s \in \{0, 1\} \qquad s \in \mathcal{S} \tag{A.120}$$

$$y_{ct} \in \{0, 1\} \qquad c \in \mathcal{C},\ t \in \mathcal{T} \tag{A.121}$$

$$\delta^{+}_{ct},\ \delta^{-}_{ct} \geq \mathbb{N}_0 \qquad t \in \mathcal{T},\ c \in \mathcal{C} \tag{A.122}$$

$$\mu_{ct} \geq \mathbb{N}_0 \qquad c \in \mathcal{C},\ s \in \mathcal{S},\ t \in \mathcal{T} \tag{A.123}$$

Appendix B

Demand Structure of Test Problems

Figure B.1: Demand structure of P1-8w-9e

Figure B.2: Demand structure of P2-10w-6e

Figure B.3: Demand structure of P3-4w-45e

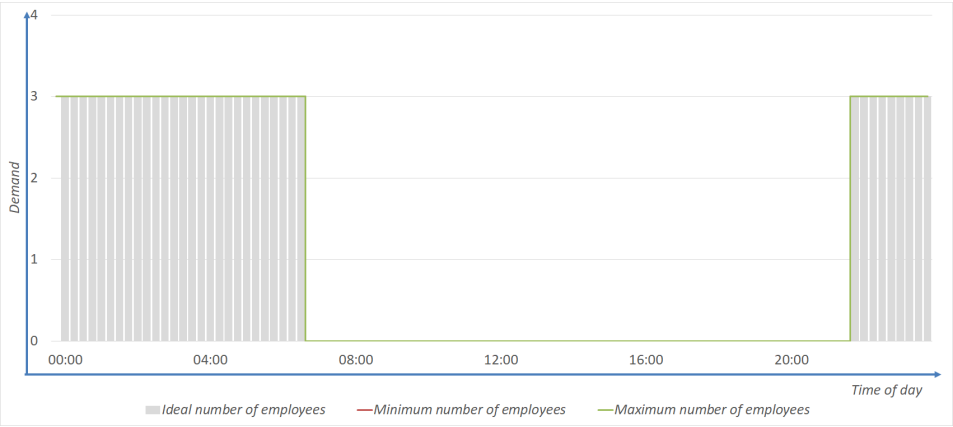

Figure B.4: Demand structure of P4-12w-8e

Figure B.5: Demand structure of P5-6w-15e

Figure B.6: Demand structure of P6-12w-10e

Figure B.7: Demand structure of P7-12w-18e

Figure B.8: Demand structure of P8-2w-13e

Figure B.9: Demand structure of P9-12w-18e

Appendix C

Supplementary Figures for Computational Study

C.1 Supplements to Technical Analysis of the PALNS

C.1.1 Number of Subprocesses

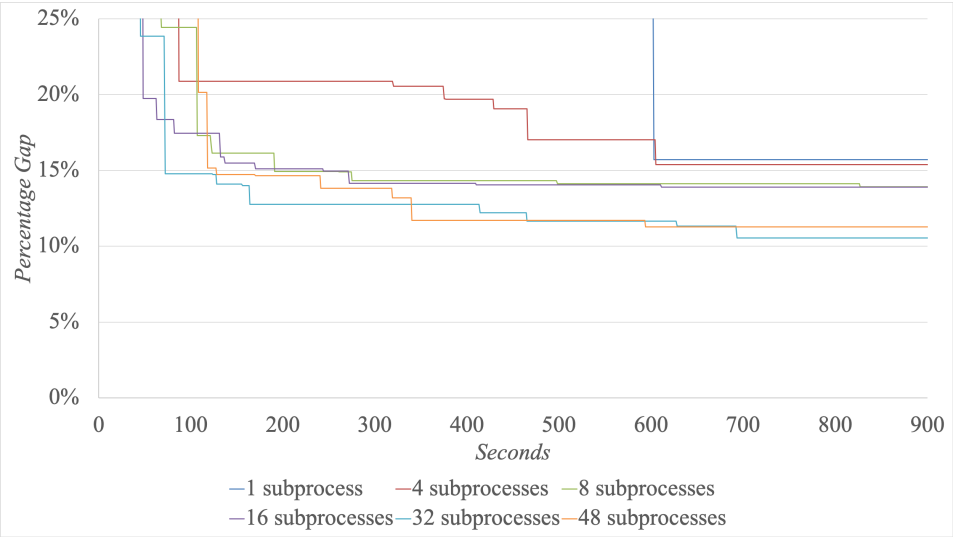

Figure C.1: Development for different number of subprocesses for P3-4w-45e

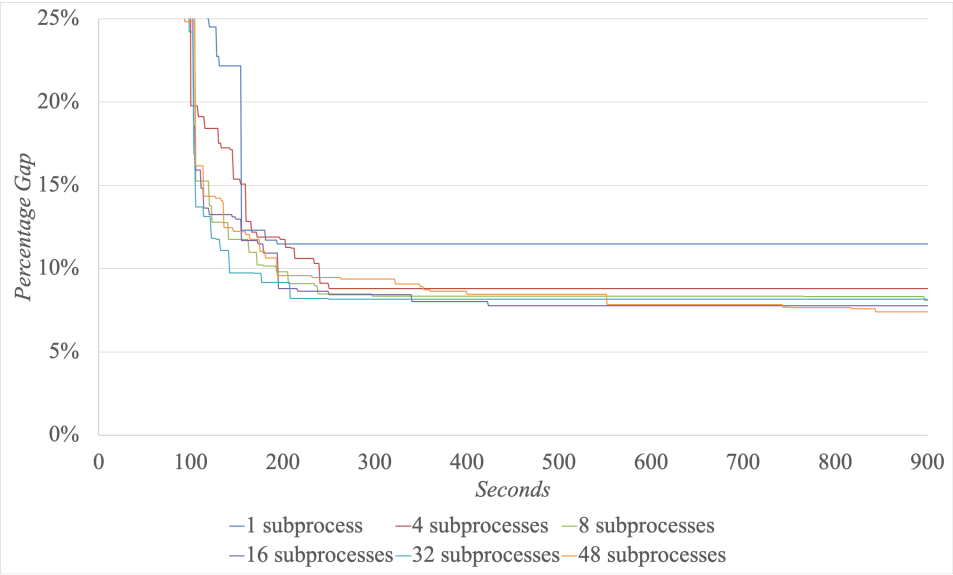

Figure C.2: Development for different number of subprocesses for P5-6w-15e

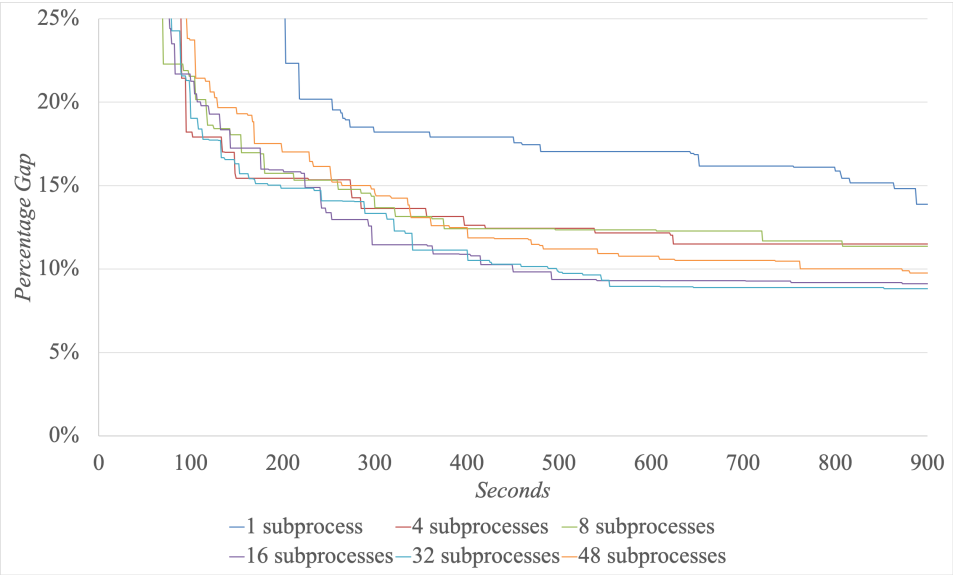

Figure C.3: Development for different number of subprocesses for P6-12w-10e

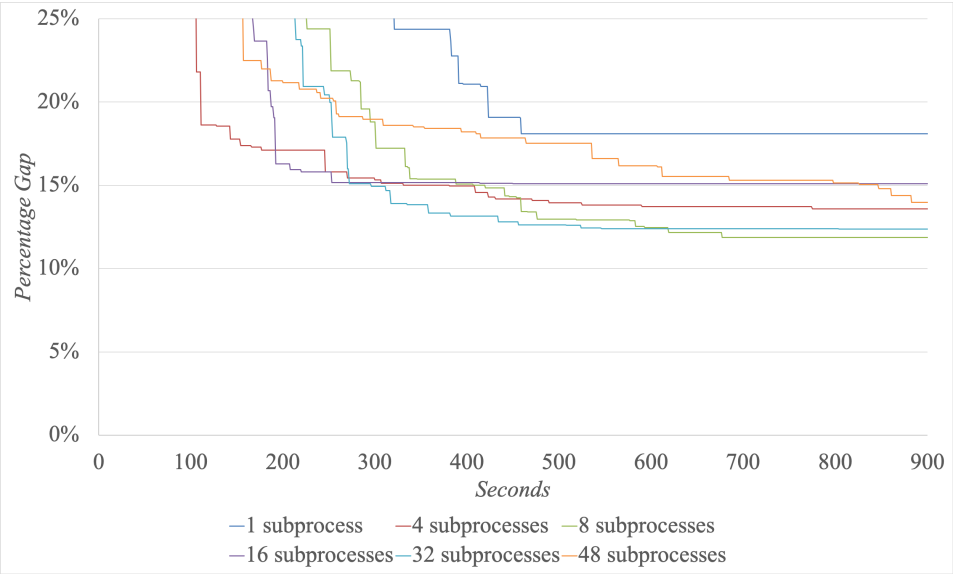

Figure C.4: Development for different number of subprocesses for P9-12w-18e

C.1.2 Acceptance Criteria

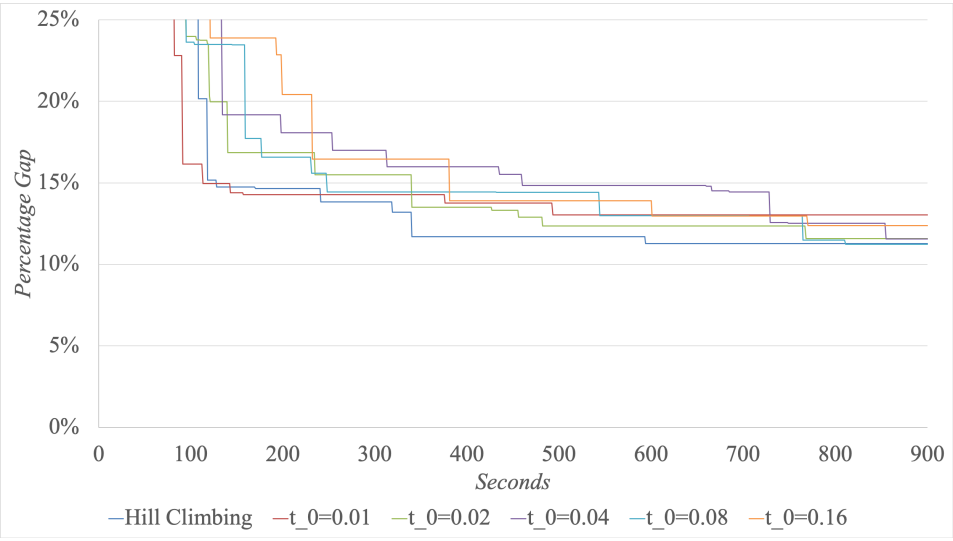

Figure C.5: Development of Hill Climbing and Record to Record Travel with threshold of 1%, 2%, 4%, 8% and 16% for P3-4w-45e

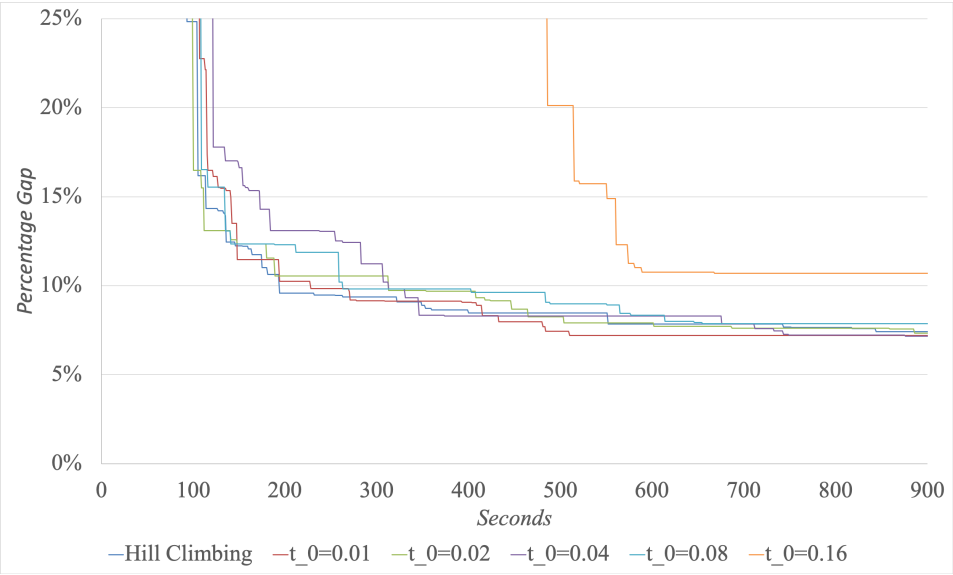

Figure C.6: Development of Hill Climbing and Record to Record Travel with threshold of 1%, 2%, 4%, 8% and 16% for P5-6w-15e

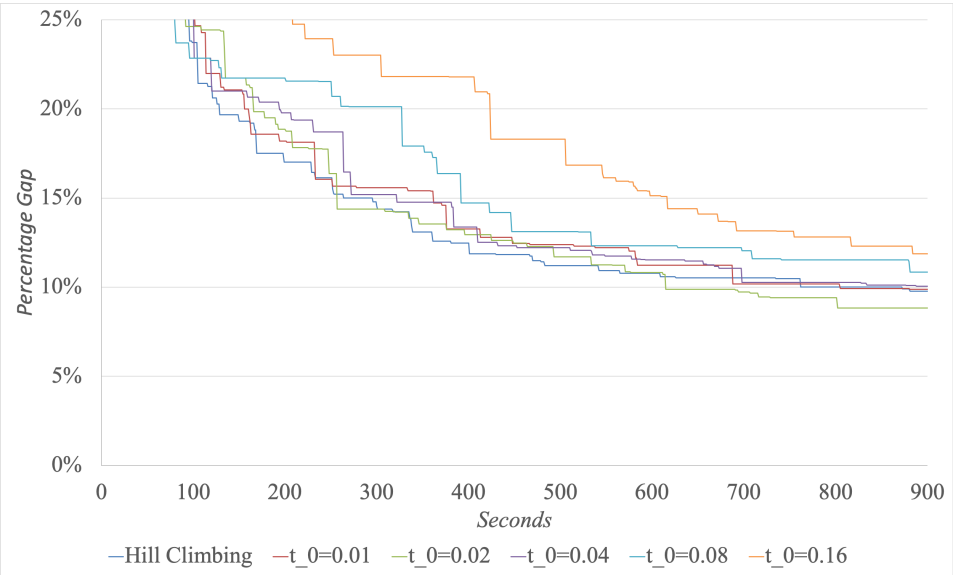

Figure C.7: Development of Hill Climbing and Record to Record Travel with threshold of 1%, 2%, 4%, 8% and 16% for P6-12w-10e

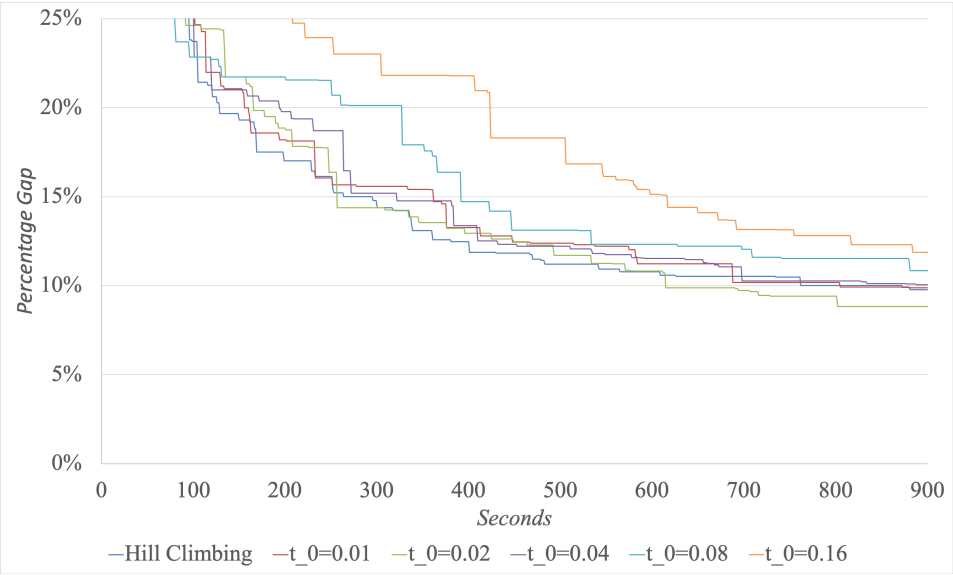

Figure C.8: Development of Hill Climbing and Record to Record Travel with threshold of 1%, 2%, 4%, 8% and 16% for P7-12w-18e

C.2 Supplements to Evaluating the Implemented Solution Approaches for the Medvind Scheduling Problem

Figure C.9 and C.10 shows how the solution develops over time for the test problems with MIP model run with and without the SR extension. Note that two separate graphs are provided for readability reasons, as some of the problems are solved faster than others. The dashed lines represent the development for the MIP model when the SR extension is applied, and is to be compared to the solid line in the same colour, representing the development for the MIP model without the SR extension for the respective problem. Figure C.9 shows the development over 15 minutes for problem P1-8w-9e, P2-10w-6e, P4-12w-8e and P8-2w-13e, while Figure C.10 shows the development over 60 minutes for problem P3-4w-45, P6-12w-10e and P7-12w-18e. A logarithmic scale is applied to the horizontal axis of Figure C.9 in order to better visualize the development. Problem P5-6w-15e are not included as the SR extension for this problem has no effect, and a feasible solution are just barley found for problem P9-12w-18e within the time limit of five hours, and this problem are thus not considered relevant to include.

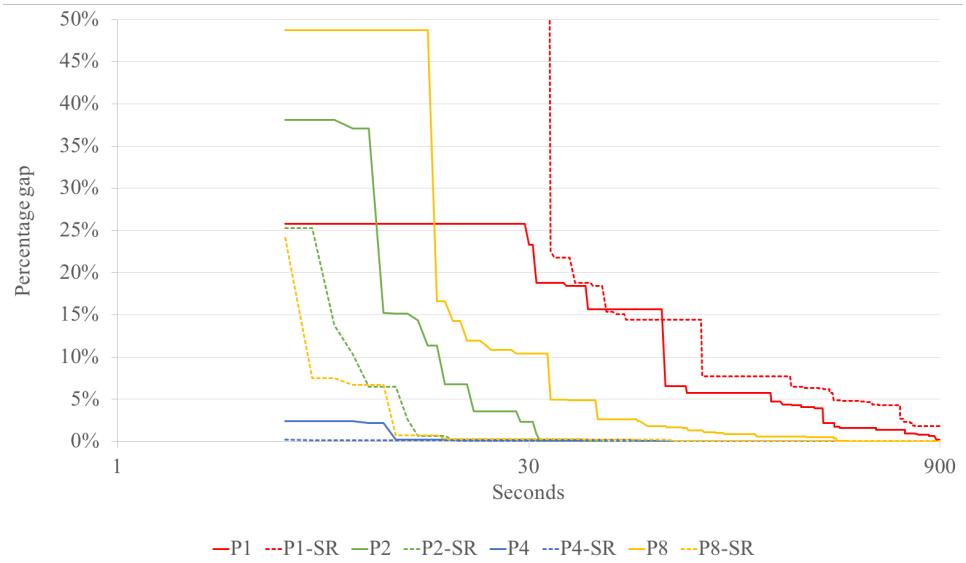

Figure C.9: The development of the percentage gap over 15 minutes for selected problems

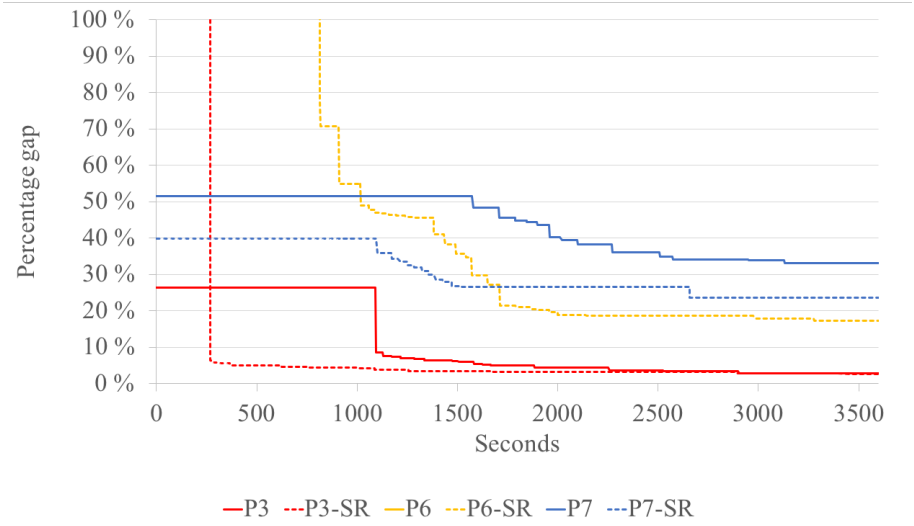

Figure C.10: The development of the percentage gap over 60 minutes for selected problems

NTNU

Norwegian University of
Science and Technology