Aalto University
School of Science
Master's Programme in Mathematics and Operations Research

Aleksi Porokka

Train driver rostering in Finland considering driver satisfaction

Master's Thesis
Espoo, September 8, 2017

Supervisor: Professor Ahti Salo
Advisor: D.Sc. (Tech.) Antti Punkka

Aalto University
School of Science

Aalto University
School of Science
Master's Programme in Mathematics and Operations Research

ABSTRACT OF MASTER'S THESIS

Author:	Aleksi Porokka		
Title: Train driver rostering in Finland considering driver satisfaction			
Date:	September 8, 2017	**Pages:**	51
Major:	Systems and operations research	**Code:**	SCI3055
Supervisor:	Professor Ahti Salo		
Advisor:	D.Sc. (Tech.) Antti Punkka		

Train driver rostering is the process of creating work timetables for train drivers. The train drivers are assigned to work shifts which usually consist of multiple tasks, to meet the service demand of the organization. The final rosters have to fulfill legal and contractual requirements, as well as take the work well-being of the drivers into consideration.

This thesis describes a real-world application of train driver rostering. The rostering problem faced by the Finnish state-owned railway company, VR Group, is modeled as a network and solved using a novel heuristic approach. The main goals of the rostering is to improve the utilization rate of the drivers and to distribute the strenuous work tasks evenly among the drivers.

The algorithm operates in three phases and relies heavily on shift removal and insertion heuristics, which are used to guide the search. The algorithm's performance is analyzed using three real-world test problems. The test results are of high quality and display the capabilities of the solution approach.

The optimization approach was taken into production soon after the development was finished. The practical effects of the automatized solution have been substantial. The utilization rate has improved, while taking the work well-being of the drivers into consideration in a more structured manner. Also, the planning time has reduced considerably.

Keywords:	Crew rostering, heuristic, rail transportation, personnel scheduling, network model
Language:	English

Aalto-universitetet
Högskolan för teknikvetenskaper
Magisterprogrammet i matematik och operationsanalys

SAMMANDRAG AV DIPLOMARBETET

Utfört av:	Aleksi Porokka		
Arbetets namn: Skapandet av jämnställda tjänstgöringsscheman för tågförare i Finland			
Datum:	Den 8 September 2017	**Sidantal:**	51
Huvudämne:	System- och operationsanalys	**Kod:**	SCI3055
Övervakare:	Professor Ahti Salo		
Handledare:	TkD Antti Punkka		

Skapandet av tjänstgöringsscheman för tågförare handlar om att tilldela arbetsskift till arbetare. Alla arbetsskift i organisationen måste fyllas av någon arbetare. De planerade tjänstgöringsscheman måste uppfylla lagliga och kontraktsbaserade krav samt ta arbetarnas välbefinnande i beaktande.

Denna avhandling beskriver en tillämpning av planering av tjänstgöringsscheman för tågförare. Planeringen har varit en utmaning för det finska, statligt ägda järnvägsbolaget VR Group. Problemet modelleras som ett nätverk och löses med hjälp av en ny heuristisk metod. Det huvudsakliga målet med den automatiserade lösningen är att förbättra användningsgraden av förarna och att distribuera de ansträngande arbetsuppgifterna jämnt bland förarna.

Algoritmen fungerar i tre skeden och bygger starkt på heuristik som tar bort arbetsskift och lägger dem tillbaka med syftet att förbättra lösningen. Algoritmens prestanda analyseras med hjälp av tre verkliga testproblem. Testresultaten, som är av hög kvalitet, visar bra vad som är möjligt med algoritmen.

Optimeringsmetoden togs i användning snabbt efter att utvecklingen var färdig. De praktiska effekterna av den automatiserade lösningen har varit betydande. Utnyttjandegraden har förbättrats, samtidigt som arbetets välbefinnande tas i beaktande på ett mer detaljerat sätt. Även tiden som krävs till planeringen har minskat avsevärt.

Nyckelord:	Tjänstgöringsschema, heuristik, tågtransport, nätverk, personal
Språk:	Engelska

Contents

Chapter 1

Introduction

1.1 Background

VR Group (VR, for short) is a Finnish state-owned railway company. It employs around 8000 people and has a turnover of about 1200 million euros.[1] Out of its employees, approximately 900 people work as train drivers in freight and long distance train traffic. These drivers have irregular work times and regulations that determine feasible work schedules they can follow. The efficient use of the drivers is essential for producing high-quality services at a competitive price.

The planning of the work schedules for the drivers starts with the planning of work shifts. After the shifts have been planned, they are assigned to the drivers to produce work rosters. The rosters are three week long work schedules that have to be planned up to one minute accuracy. Traditionally the rosters are planned manually, but there are several challenges with manual planning. First, finding high quality solutions from the large set of possible shift combinations is difficult and requires problem-specific knowledge. Second, it has been difficult to modify the rosters manually in case of sudden changes, because small changes in the input data can shift the optimal solution far away from the original optimum. Third, with manual planning it is not possible to plan multiple alternative rosters for an individual depot. This would be beneficial, because it would enable the planners to emphasize different aspects in the rosters, and thus provide a set of solutions for the representatives of the train drivers' union to choose

[1] www.vrgroup.fi/en, visited August 21, 2017

from.

An automatized solution would also help planners increase the utilization rate of the drivers and take the drivers' work well-being into consideration already at the outset. There have been previous attempts to automatize the planning of the rosters at VR using a Mixed Integer Linear Program model. However, the results were not satisfactory. Thus, there was still a need for an optimization model that would perform up to the standards required by the planning department and that would have the potential to include multiple driver-specific requirements in the future.

1.2 Research objectives

The objective of this thesis is to develop an optimization model for train driver rostering. The rosters, which have to fulfill legal and contract based requirements, consist of two types of train drivers: "regular drivers" and "extra drivers". The extra drivers can be used flexibly during the work period to perform work on short notice, whereas the rosters for the regular drivers cannot be modified after the rosters have been published. Thus, the utilization rate of the regular drivers should be maximized in the rosters to minimize the amount of work planned for the extra drivers.

The work well-being of the drivers should be supported by ensuring sufficient rest periods during the work periods and by distributing strenuous shifts evenly among the drivers. The distribution needs depend on the depots, because the depots have different amounts and types of strenuous shifts.

The optimization model should produce feasible solutions quickly as the rosters for the next work period have to be planned in a few days. Also, having the possibility to produce new solutions after sudden changes and after receiving driver feedback would make the planning process more robust, increase driver satisfaction and improve the dialogue between the employer and the employees.

Prior to this thesis the rosters have been planned manually using a proprietary software. The integration of the software and the optimization model was not a focus of this thesis. The integration may however become relevant in the future in case the optimization model proves to be successful in practice as the software is used company wide and the final rosters are stored there.

1.3 Structure of the thesis

The remaining part of the thesis is structured as follows. A literature survey of personnel scheduling with focus on crew rostering is given in Chapter 2. In Chapter 3 the train driver rostering problem of VR is presented in detail and an optimization model is formulated. The optimization algorithm is introduced in Chapter 4 and the results are presented in Chapter 5. In Chapter 6 practical experiences from using the automatized rostering approach are discussed. Finally, ideas for future research are discussed in Chapter 7, which also concludes the thesis.

Chapter 2

Literature survey

The number of publications on train driver rostering in the literature is not large, while there has been more work on other personnel rostering applications, such as nurse rostering and airline crew rostering. Section 2.1 contains an overview of personnel rostering and the abovementioned application areas are presented in Sections 2.2 – 2.4.

2.1 Personnel scheduling

Personnel scheduling involves creating work timetables for the staff to satisfy the demand of the goods or services provided by the organization (Ernst et al., 2004). This can be viewed as the process of assigning staff members to shifts to meet the service demand. The resulting rosters have to fulfill workplace agreements, which set rules for the scheduling process. The competences of the individual staff members have to be taken into consideration, because certain shifts may require skills not everyone has. To increase the work well-being of the personnel further constraints may be imposed (Van den Bergh et al., 2013). For example, the organization may want to try to fulfill the day off requests of the staff members, or try to take personal preferences, such as preferences concerning night and day shifts, into consideration. An example roster from VR can be seen in Figure 2.1.

Personnel scheduling can roughly be divided into cyclic and non-cyclic scheduling (Ernst et al., 2004). In cyclic scheduling, the same rosters are used for multiple scheduling periods, whereas in non-cyclic scheduling new rosters are created for each new scheduling period separately (Cheang et al., 2003). At VR, there has been a history of using both cyclic and non-cyclic scheduling, but the importance of non-cyclic scheduling is increasing in the company. Non-cyclic rostering makes it easier to take period-specific constraints into account, such as changes in the shifts or in the availability of the workers.

Personnel rostering methods have been utilized in multiple lines of business. Mehrotra (1997) and Grossman et al. (1999) provide reviews discussing the use of operations research techniques for call centre problems. Talarico and Duque (2015) describe an application of personnel scheduling for a retail chain. Rasmussen et al. (2012) present the problem of allocating home carers to patients' home with the goal of maximizing the overall service level. Li and Womer (2009) address the crew scheduling problem on a ship where the crew is required to handle various onboard tasks. An optimization problem where schedules need to be created for Australian navy boats and their crews is presented by Horn et al. (2007). Sabar et al. (2009) discuss the scheduling of personnel to work stations in an assembly center. A personnel scheduling problem where the workforce needs to be assigned to check-in counters at airports is described by Stolletz (2010). Qi and Bard (2006) develop a simulation model for solving personnel scheduling challenges related to mail handlers. Airline crew and nurse rostering are among the most common practical implementation areas of personnel rostering. These subjects are discussed more thoroughly in the subsections 2.2 and 2.3.

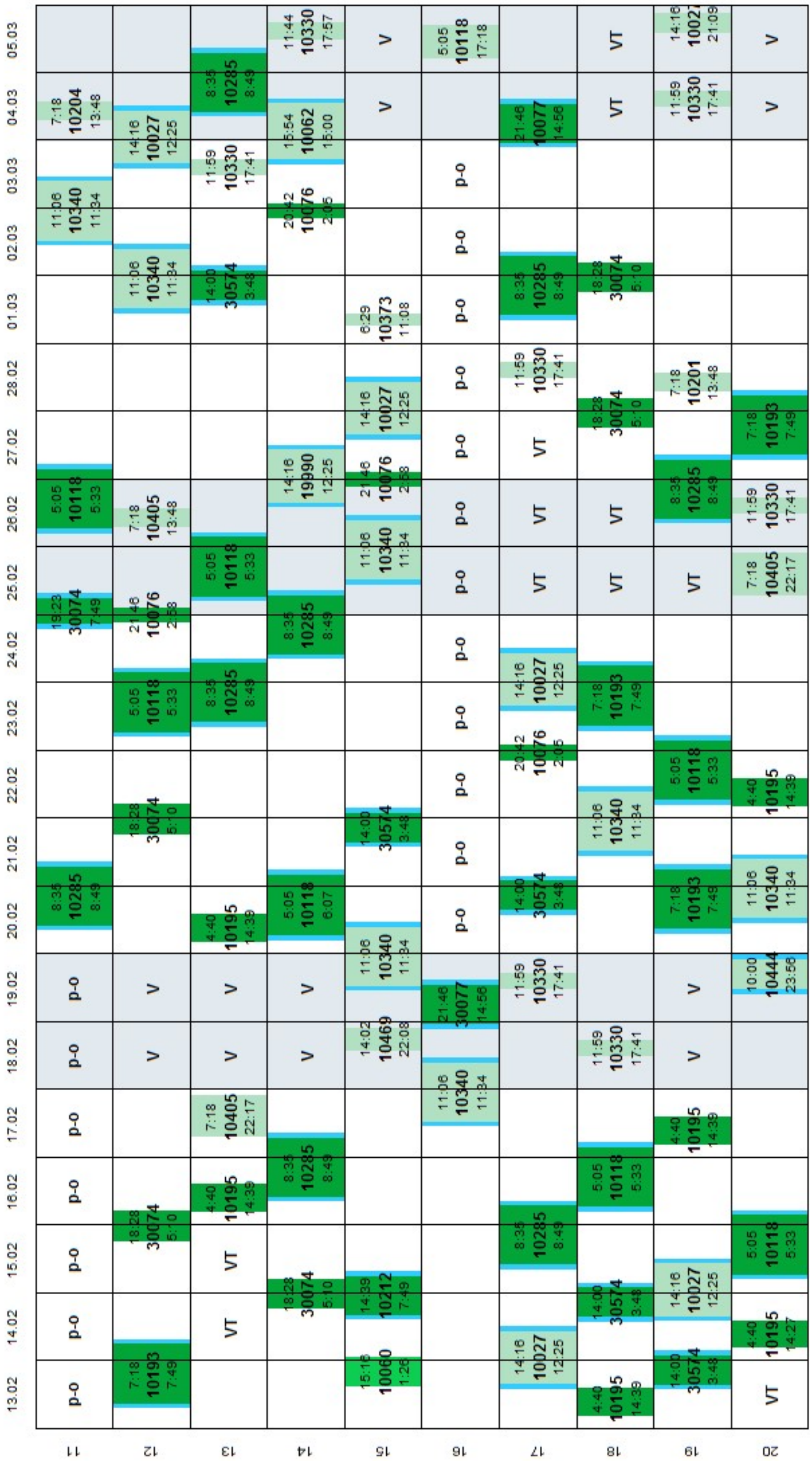

Figure 2.1: An example roster created for a group of train drivers at VR. Each row corresponds to a driver's work schedule. The shifts are marked as boxes with different colours to represent the various attributes of the shifts. "p-o", "V" and "VT" are codes used to represent the drivers' absences.

2.2 Nurse rostering problem

Many publications discuss nurse rostering problems. The nurse rostering problem, also known as the nurse scheduling problem, involves assigning nurses to shifts. The review by Burke et al. (2004) describes the typical variants of the nurse rostering problem as well as the methods for solving them. The authors present detailed tables of various constraints and objectives found in nurse rostering problems and also the papers which have included them in their models. The hard constraints can, for example, determine the minimum number of consecutive days off (Warner, 1976; Aickelin and Dowsland, 2000), or whether or not there has to be a free day after a night shift (Ikegami and Niwa, 2003; Bellanti et al., 2004).

Burke et al. (2004) divide the methods for solving the nurse rostering problems into mathematical programming, multi-criteria approaches, artificial intelligence methods, heuristics, and metaheuristic scheduling. They describe mathematical programming approaches suitable for finding optimal solutions, but state that usually these approaches cannot handle complex real world models. Millar and Kiragu (1998) utilize the fact that in a special case of the nurse rostering problem the nurses' schedules are made up of alternating sequences of work and off days. They formulate the problem as a network problem, which is in essence a shortest path problem with side constraints, and solve it using IBM ILOG CPLEX Optimizer.

Artificial intelligence methods include constraint programming, as well as expert systems (Burke et al., 2004). Burke et al. (2004) describe expert systems as methods that utilize the knowledge of the specific domain's experts to provide interactive decision support. Abdennadher and Schlenker (1999) present an application of constraint programming to the nurse rostering problem. Their semi-automatic rostering tool utilizes constraint programming to provide decision support to the end users.

Heuristics and metaheuristics can be applied to nurse rostering problems to obtain high quality solutions in a relatively short time (Burke et al., 2004). Ikegami and Niwa (2003) and Bellanti et al. (2004) take advantage of tabu search for creating the rosters, whereas Aickelin and Dowsland (2000) use genetic algorithms. Aickelin and Dowsland (2000) describe a genetic algorithm that utilizes problem specific knowledge of a nurse rostering problem to overcome issues related to conflicting objectives and constraints, which are commonly found in nurse rostering problems.

From a managerial perspective, the rostering of hospital staff can be performed using three approaches: departmental rostering, team rostering, and self-rostering. In departmental rostering, the responsibility of planning the rosters is given to a single person, whereas in team rostering the staff is split up into teams, which plan rosters for themselves. The teams have a specific person that is responsible for planning the rosters with the help of the rest of the team members. Self-rostering enables the individual staff members to plan the schedules for themselves. Usually the rosters resulting from team rostering and self rostering need to be accepted by a senior manager. The preferred rostering approach depends on the complexity of the problem. For simple problems, self-rostering seems to produce high quality results. With increasing complexity one should choose team rostering and departmental rostering (Silvestro and Silvestro, 2000).

Silvestro and Silvestro (2000) list several aspects of the nurse rostering problem that affect the problem complexity. Among these are the staff size, as well as the amounts of work-specific skills that have to be taken into consideration. The complexity is also affected by how well the personnel demand can be predicted and by the demand variability.

2.3 Airline crew rostering

Airline crew rostering is the process of assigning crew pairings and other tasks, such as ground shifts, to rosters. In crew pairing, a sequence of flights are joined together and for each flight in the pairing all the crew requirements have to be satisfied. There are many different strategies for creating rosters from the crew pairings. The bidlines approach starts by creating anonymous rosters, which the crew members bid on. Based on these bids, the anonymous rosters are assigned to the crew members. The European airlines favor strategies that create directly personalized rosters, which can take personal preferences and quality criteria into consideration. Even though the different strategies approach the airline crew rostering problem from different perspectives, the models are quite similar and the key differences are found in the problem objectives (Kohl and Karisch, 2004).

The focus in crew pairing is usually to create efficient pairings that minimize costs, whereas in crew rostering work well-being is also a relevant factor, which may be one reason why there been more research conducted on crew pairing than on rostering. The typical constraints for the airline crew rostering usually originate from law requirements, or agreements between the airline and employee unions. The constraints can be divided into three categories: horizontal, vertical and artificial rules (Kohl and Karisch, 2004).

Horizontal rules are specific for each crew member. For example, some shifts can only be assigned to crew members with the right qualifications. Vertical rules depend on the attributes of multiple crew members. For example, in some cases it can be infeasible to assign two inexperienced crew members to the same pairing. Artificial rules originate usually from the airline itself. The airline may for example wish to create robust solutions, which do not easily become infeasible in case of delays or other changes.

Gopalakrishnan and Johnson (2005) list three reasons why the airline crew rostering problem can be difficult to solve:

1. The number of crew pairings can be extremely large. For example, for a moderate size fleet the number of pairings can be 100 million, whereas for the large North American fleets the number of pairings can be tens of times larger.
2. There are complex safety regulations and work rules that have to be satisfied.
3. Crew costs are nonlinear and difficult to model.

Multiple solution approaches have been applied to the airline crew rostering problem. Campbell et al. (1997) apply the bidlines method at FedEx, and Christou et al. (1999) describe a bidlines application developed for Delta Air Lines. Kohl and Karisch (2004) present a method based on set partitioning for the personalized crew rostering problem. Cappanera and Gallo (2001) apply mathematical programming methods to the airline crew rostering problem, where the problem is formulated as a 0-1 multi-commodity flow. Dawid et al. (2001) and König and Strauss (2000a,b) describe heuristic algorithms that have been applied to a few European airlines.

2.4 Train and bus crew rostering

Railway crew management can be split into two categories, namely crew scheduling and crew rostering. Crew scheduling is the process of creating shifts that cover all the train trips. In crew rostering these shifts are then assigned to the crews. Caprara et al. (1997) present three reasons for decomposing the railway crew management into crew scheduling and crew rostering

1. Crew scheduling involves constraining the starting and ending locations of the shifts based on the home depots of the crews. Thus, it seems natural to first create shifts for the depots, and then create rosters for the depots independently.
2. The constraints for crew scheduling are of different type than those used for crew rostering. The authors clarify this with examples from an Italian railway company: the crews have to have at least a few minutes of spare time to change trains between two consecutive trips in a shift, whereas the rest time between two consecutive shifts has to be between 18 and 22 hours. Thus the required rest times are of completely different orders of magnitude.
3. Combining and solving the two problems together is extremely difficult, because even the individual problems are challenging.

The objective of railway crew management is generally to minimize the number of crews needed to cover the trips of the work period. The final costs depend both on the shifts created in crew scheduling and the rosters planned during crew rostering. Clearly, as the problems are inherently related to each other, one has to take the rostering problem into consideration already during scheduling (Caprara et al., 1997).

Caprara et al. (1997) approach a real-world crew rostering problem of the Italian railways using a construction heuristic that utilizes the solution of a relaxed problem. The construction heuristic and the relaxation are presented in detail by Caprara et al. (1998). The length of the rosters described by Caprara et al. (1997) are generally between 30 and 60 days. The shifts have multiple attributes. For example the shifts can contain a rest, and there are two types of night shifts depending on when the night work occurs. There are also two types of work times, out of which one contains additional paid time. The constraints are typical to the problem type. Examples of the constraints included in the model are upper limits for the total work time during the work period and lower bounds for the minimum number of week rests. Their solution approach is able to find the optimal solution to 6 out of 7 test problems. Solving the largest problem with 525 shifts took 1185 seconds using a PC with a Pentium 90 CPU.

Bianco et al. (1992) describe a heuristic algorithm for solving the crew rostering problems of mass transit systems. Their approach is based on a heuristic that solves a multi-level bottleneck assignment problem at each iteration. In addition to the shift covering constraints, the model includes constraints that restrict the minimum rest time between two consecutive shifts. The model also aims at balancing the total work times of the drivers. The algorithm solves problems with up to 130 shifts and a planning period of seven days.

Borndörfer et al. (2017) integrate the shift scheduling and rostering parts for a public transport application. They integrate the problems using shift templates, instead of individual shifts, to reduce the problem complexity. They solve the model using an approach based on Benders decomposition. Valdes and Andres (2010) present a set covering model used to solve the crew scheduling and crew rostering problems simultaneously. The test instances are based on real-world bus systems.

Xie and Suhl (2015) present a crew rostering model for public bus transit. The model, which is formulated as a multi-commodity network flow, can be used to solve both cyclic and non-cyclic crew rostering problems. Yunes et al. (2005) compare the hybrid column generation algorithms to mathematical programming and constraint logic programming approaches for solving the crew scheduling and rostering problems faced by urban bus transit. In their experiments, which are based on real-word data from Belo Horizonte, Brazil, the hybrid algorithms are able to find optimal solutions for more problem cases than the solution methodologies solely based on mathematical programming or constraint logic programming.

Chapter 3

Problem formulation

In this section the characteristics of the train driver rostering problem are discussed. The problem constraints are first presented using examples to clarify the concepts. Some background information is also given to show the importance and origins of the constraints. Based on the informal description, a network model of the problem is formulated in Section 3.2.

3.1 Problem description

The train driver rostering problem is solved independently for each of the 25 depots in Finland. Each train driver has a home depot, where his/her shifts have to start from and end at. As each work shift is assigned to a depot, the rostering problem can be solved depot by depot.

The problem includes both soft and hard constraints. Hard constraints, which originate from union agreements and law requirements, define the feasibility of a solution, whereas the soft constraints affect the solution quality and aim at increasing the work well-being of the drivers.

3.1.1 Hard constraints

The work time that includes the work time compensations is called *artificial work time*, whereas *real work time* does not include work time compensations. Drivers earn work time compensations from shifts that contain evening or night work: for every hour a shift contains work between 21.00 and 06.00 the drivers get twenty minutes of work time compensations in addition to the real work time. Also, if the shift starts at 04.00 or earlier the drivers earn similar work time compensations until they have an at least two hour long continuous break, or until the clock strikes 12.

In each three week work period the drivers cannot have more than 114 hours and 45 minutes of artificial work time. For periods that contain public holidays, such as Christmas or Easter, the artificial work time limit is lower. The artificial work hour limit is driver-specific, because vacations and other absences also affect the limit. The maximum amount of night work, i.e., work between 22.00 and 06.00, is limited to 42 hours during the work period. This limit is not affected by public holidays.

The maximum number of real work hours in a *work cluster*, which is the period of work between two consecutive *double week rests*, is limited to 45 hours for each driver. Double week rests are defined as at least two consecutive calendar days that do not contain any work, i.e., no shifts end or start during the day. The maximum number of calendar days between two double week rests is five. If, for example, a driver has a double week rest on the weekend and work on Monday, Tuesday, Thursday and Friday, then he has been working on five consecutive work days on Friday, even though there would not be work on Wednesday. Thus he has to have a double week rest on the coming weekend.

The shifts can be classified as *day* or *night shifts*. Shifts that are not classified as night shifts are considered as day shifts. Nights shifts can be divided into two classes. Night shifts of type B contain work between 02.00 and 05.00 during a night, whereas night shifts of type A contain at least three hours of work between 22.00 and 06.00 during a single night. The number of consecutive nights with *night work* (the night contains a shift that has work of type A or B during the night) a driver can be assigned to is constrained to two. However, a driver cannot have nights that contain night work of type B on consecutive nights.

The drivers can only be assigned to shifts they have proper training for. The drivers cannot be assigned to shifts that that contain work during agreed vacations or absences. These constraints are driver-specific, because they depend on the skills and schedules of the drivers.

To summarize, a feasible 3-week roster satisfies the following hard constraints

HC1. Driver-specific upper bound on artificial work time during the work period (usually 114 hours and 45 minutes)

HC2. At most 42 hours of night work during the work period

HC3. At most five calendar days between two consecutive double week rests

HC4. No more than 45 hours of between two consecutive double week rests

HC5. At least ten hours of rest between two consecutive shifts

HC6. No night shifts of type B on consecutive nights

HC7. Three consecutive nights with night shifts is not allowed

HC8. The drivers can only be assigned to shifts for which they have the required skills

HC9. The drivers cannot be assigned to any shifts during absences

HC10. Each shift is assigned to a driver

3.1.2 Soft constraints

Work hours between 00.00 and 24.00 on Sundays, church holidays, on the Independence Day, or on May day are classified as *Sunday work*. The work hours between 18.00 and 24.00 on the previous day are also considered as Sunday work. For every Sunday work hour the drivers get a higher salary, which is the main reason why it is desirable to constrain the amount of Sunday work a single driver can have and to divide the Sunday work more equally between the drivers.

Shifts with rest are shifts that contain a period of time inside the shift itself that is not counted as work time, and the driver only gets a fixed monetary compensation for each rest. Such rest time is usually spent away from the home depot, which together with the lower salary, make it relevant to limit the maximum number of shifts with rest that a driver can be assigned to.

Night shifts are considered more strenuous than day shifts, which is why distributing the night shifts evenly among the drivers is relevant. In the night shift distribution rules no distinction is made between night shifts of type A and B.

To summarize, a preferred roster takes the following aspects into consideration

SC1. Balance the amount of Sunday work evenly among the drivers

SC2. Balance the number of night shifts evenly among the drivers

SC3. Balance the number of shifts with rest evenly among the drivers

3.1.3 Objectives

The main objective of the rostering process is to increase the utilization rate of the regular drivers by maximizing the number of artificial work hours planned for them. A secondary objective is to make the rosters compact. In a *compact roster* the rest between two consecutive shifts should be as close to ten hours as possible if there is not a double week rest between the shifts. The number of work clusters should also be minimized. This is because most drivers prefer having as much work as possible in a work cluster, which enables them to have many consecutive days off during the double week rests.

3.2 Formal definition

The train driver rostering problem is modeled using a shift graph. Let the set S contain m shifts, which all start during the work period that is n_d days long. The shifts do not need to end during the work period: they can continue until the first day of the next period, i.e., day $n_d + 1$. The starting and ending day for shift $i \in S$ can thus be denoted by $s_d(i) \in \{1, ..., n_d\}$ and $e_d(i) \in \{1, ..., n_d + 1\}$, correspondingly. To store the starting and ending minute of the shift we define a set F so that it contains all the minutes in the work period. As each day contains 1440 minutes and the work period is n_d days long, $F = \{1, 2, ..., 1440n_d\}$. The starting minute of shift i is denoted by $s_m(i) \in F$ and the ending minute by $e_m(i) \in F \cup \{1440n_d + 1, ..., 1440(n_d + 1)\}$.

In the shift graph $G = (S, E)$ each shift is represented by a node. The edge (i, j) between the nodes $i, j \in S$ belongs to E if and only if $e_m(i) - s_m(j) \geqslant \Delta$, where $\Delta = 600$ is the minimum rest time between two consecutive shifts (HC5). The set of nodes that can be reached from node i is denoted by $A(i) = \{j \in S : (i, j) \in E\}$. The graph is directed and acyclic, because the starting times define an order in which the shifts have to be completed. A path P denotes a sequence of nodes in the shift graph G, which can be seen from Figure 3.1. Each path will be assigned to a driver d, which is why driver-specific information needs to be taken into consideration when creating the paths, thus $P = P(d)$. In a feasible roster R each shift must be assigned to one path P that belongs to R (HC10).

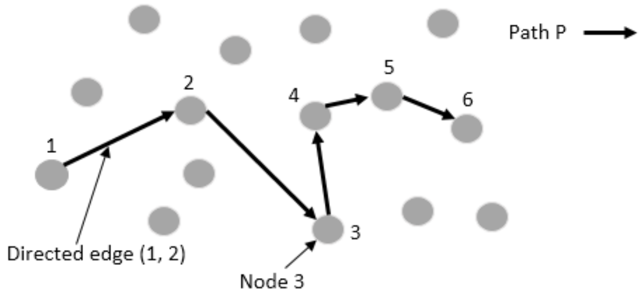

Figure 3.1: The figure displays a shift graph that consists of nodes and edges. The path P is a sequence of directed edges $((1,2),(2,3),(3,4),(4,5),(5,6),(6,7))$ between the nodes. The directed edges exist only if the resting time between the the nodes, i.e., shifts, is at least ten hours (HC5).

3.2.1 Night shift constraints

The time difference between the starting time and the ending time of a shift cannot exceed 24 hours. Thus a shift can contain work on at most two consecutive nights. The work period contains n_d days, but $n_d + 1$ nights. A night is defined as the period between 22:00 and 06:00 on the following day. However the first night corresponds to the hours 00:00 - 06:00 on day one. Let $H = \{1, ..., n_d + 1\}$ denote the set of all the nights. Each night

$h \in H$ is defined by a starting minute $s_m(h)$ and ending minute $e_m(h)$. The minutes shift i spans is denoted by $F(i) = \{s_m(i), ..., e_m(i)\}$. In a similar manner we denote the minutes night h spans by $F(h) = \{s_m(h), ..., e_m(h)\}$. To define nights of type B we have a set $F_B(h) \subset F(h)$ that contains the minutes belonging to the time interval 02:00 - 05:00 on night h. We can determine the type of night work of shift i during night h in the following manner

(a) We define $H(i) \subset H$ to contain the nights h for which shift i has work on, i.e., $|F(i) \cap F(h)| \geqslant 1$ holds. Further, let $H_{AB}(i) \subseteq H(i)$ be the nights i has night work of type A or B on, and $H_B(i) \subseteq H_{AB}(i)$ as the nights i has night work of type B on. Now if $F(i) \cap F_B(h) \neq \emptyset$, then i contains h as type B. Thus $h \in H(i)$, $h \in H_{AB}(i)$, and $H_B(i) = \{h\}$.

(b) If $|F(i) \cap F(h)| \geqslant 180$ and $h \notin H_B(i)$, then $h \in H(i)$ and $h \in H_{AB}(i)$.

If neither (a) or (b) holds then i does not contain night work of any type on night h. A shift cannot contain two nights of type B, and thus $|H_B(i)| \leqslant 1$. On the other hand, as a shift cannot span more than two nights $|H(i)| \leqslant 2$ holds for all shifts i.

It is noteworthy to remember that a single night h of a single driver cannot contain two shifts during one night because the break between two shifts has to be at least ten hours long, i.e., $\Delta = 600$.

To express the night shift constraints we define two cumulative variables, $N_{AB}(P)$ and $N_B(P)$. The variable $N_{AB}(P)$ stores the number of consecutive nights of type A or B at the last shift of path P, while $N_B(P)$ stores the number of consecutive nights of type B at the last shift of P.

Let $h_i^{-1} \in H$ be the night before the start of shift i, that is

$$h_i^{-1} = \begin{cases} s_d(i) & \text{if } s_d(i) \notin H(i) \\ s_d(i) - 1 & \text{otherwise.} \end{cases} \tag{3.1}$$

If we add shift j after the current last shift i in the path $P \neq \emptyset$, we get $N_{AB}(P')$ and $N_B(P')$ for the new path P' in the following manner

$$N_{AB}(P') = \begin{cases} N_{AB}(P) + |H_{AB}(j)| & \text{if } h_j^{-1} \in H_{AB}(i) \text{ or } h_i^{-1} = h_j^{-1} \\ |H_{AB}(j)| & \text{otherwise} \end{cases} \tag{3.2}$$

$$N_B(P') = \begin{cases} N_B(P) + |H_B(j)| & \text{if } h_j^{-1} \in H_B(i) \text{ or } h_i^{-1} = h_j^{-1} \\ |H_B(j)| & \text{otherwise.} \end{cases} \tag{3.3}$$

Examples of the conditions $h_j^{-1} \in H_B(i)$ and $h_i^{-1} = h_j^{-1}$ are given in Figures 3.2 and 3.3. The path P' satisfies the night constraints if $N_B(P') \leqslant 1$ (HC6) and $N_{AB}(P') \leqslant 2$ (HC7).

In case j is added to an empty path $P = \emptyset$ the cumulative variables are computed as $N_{AB}(P') = |H_{AB}(j)| + p_{AB}(d, h_j^{-1})$ and $N_B(P') = |H_B(j)| + p_B(d, h_j^{-1})$, where $p_{AB}(d, h_j^{-1})$ denotes the number of consecutive night of type A or B at night h_j^{-1} for driver d, and $p_B(d, h_j^{-1})$ the number of consecutive nights of type B at night h_j^{-1} for driver d. The driver d is the that is associated with the path P'. $p_{AB}(d, h_j^{-1})$ and $p_B(d, h_j^{-1})$ are computed based on the last shifts driver d is planned to have during the previous working period.

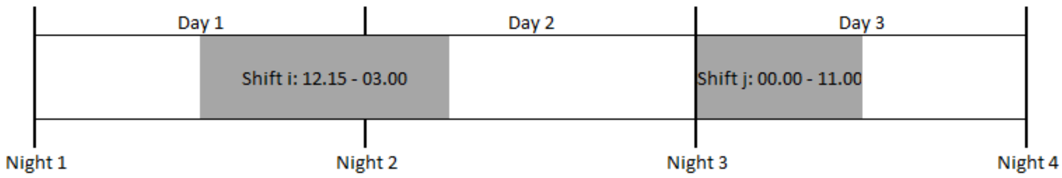

Figure 3.2: Shift i contains work of type B on the night previous to the start of shift j, i.e., $h_j^{-1} \in H_B(i)$. As j contains night work of type B on night 3, there are two consecutive nights with night work of type B on. Thus the driver cannot be assigned to both i and j (HC6).

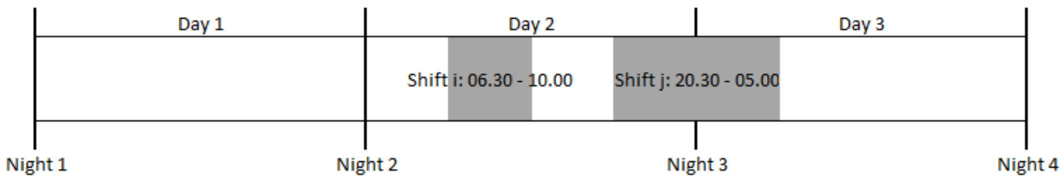

Figure 3.3: The night before the start of shifts i and j is the same night, and thus $h_i^{-1} = h_j^{-1}$. As i does not contain night work of any type, assigning i and j to the same driver does not violate the night shift constraints.

3.2.2 Work cluster constraints

A path $P = (i_1, ..., i_p)$ with p shifts spans the days starting from the starting day of the first shift $s_d(i_1)$ until the ending day of the last shift $e_d(i_p)$. Days

w that do not contain work, i.e., there does not exist a shift $i \in P$ ending on that day $s_d(i) = w$ or starting on that day $e_d(i) = w$, are known as rest days. A double week rest consists of at least two consecutive rest days. The days that are not rest days are known as work days.

For the path P two cumulative variables are used to impose the work time constraints for a work cluster:

(a) $D(P)$: the number of consecutive work days on day $e_d(i_p)$ since the last double week rest of P, or the days since the ending day of the last shift in the previous work period if $d(P)$ does not contain any double rest days.

(b) $R(P)$: the number of real work hours on day $e_d(i_p)$ since the last double week rest of P, or since the ending day of the last shift in the previous work period if $d(P)$ does not contain any double rest days.

The variables $D(P)$ and $R(P)$ are updated when a new shift j is added after the currently last shift $i \in P \neq \emptyset$, resulting in path P'

$$D(P') = \begin{cases} e_d(j) - s_d(j) + 1 & \text{if } s_d(j) - e_d(i) > 2 \\ D(P) + e_d(j) - e_d(i) & \text{otherwise} \end{cases} \tag{3.4}$$

$$R(P') = \begin{cases} t_r(j) & \text{if } s_d(j) - e_d(i) > 2 \\ R(P) + t_r(j) & \text{otherwise,} \end{cases} \tag{3.5}$$

where $t_r(j)$ is the real work time of shift j. Let the ending day of the last shift of driver d in the previous work period be $e_d(d) \leqslant 1$. The cumulative variables for the empty path $P = \emptyset$ are initialized as $D(P) = p_D(d)$ and $R(P) = p_R(P)$, where $p_D(d)$ denotes the number of consecutive work days on day $e_d(d)$ since the last double week rest, and $p_R(d)$ the number of real work hours on day $e_d(d)$ since the last double week rest. When j is added to the empty path P, the update formula 3.4 is used with the exception of $e_d(i) = e_d(d)$.

A feasible path P' satisfies the following properties: $D(P') \leqslant 5$ (HC3), and $R(P') \leqslant 45$ (HC4).

3.2.3 Work period constraints

The shifts have various attributes associated with them: artificial driving time $t_a(i)$, the amount of night work $t_n(i)$, and the amount of Sunday work $t_s(i)$. We also know if the shift is a night shift of any type, or if it contains a rest within the shift. To express the constraints concerning these attributes, we define a cumulative variable for each attribute. For path P we get the following

(a) $T_a(P)$: total accumulated artificial driving time of path P

(b) $T_n(P)$: total accumulated night driving time of path P

(c) $T_s(P)$: total accumulated Sunday time of path P

(d) $N_n(P)$: the number of night shifts on path P

(e) $N_r(P)$: the number of shifts with rest on path P.

For the empty path $P_0 = \{\}$ that does not contain any shifts we set $T_a(P_0) = T_n(P_0) = T_s(P_0) = N_n(P_0) = N_r(P_0) = 0$.

Adding a shift j to P results in path P'. The values for the cumulative variables are then given by

$$T_a(P') = T_a(P) + t_a(j) \tag{3.6}$$

$$T_n(P') = T_n(P) + t_n(j) \tag{3.7}$$

$$T_s(P') = T_s(P) + t_s(j) \tag{3.8}$$

$$N_n(P') = \begin{cases} N_n(P) + 1 & \text{if } |H_{AB}(j)| \neq 0 \\ N_n(P) & \text{otherwise} \end{cases} \tag{3.9}$$

$$N_r(P') = \begin{cases} N_r(P) + 1 & \text{if } j \text{ contains a rest} \\ N_r(P) & \text{otherwise.} \end{cases} \tag{3.10}$$

The path P' is feasible in terms of these knapsack-like constraints if the following hard constraints hold: $T_a(P') + p_{Ta}(d) \leqslant U_a(d)$ (HC1), and $T_n(P') + p_{Tn}(d) \leqslant U_n$ (HC2). Here $p_{Ta}(d)$ and $p_{Tn}(d)$ denote the number of artificial and night work hours coming from shifts that started during the last work period and ended during the current work period, respectively. The upper bound for the artificial work hours, $U_a(d)$, is driver-specific, whereas the upper bound for the night work hours is not. The soft constraints related to the Sunday time, night shifts and shifts with rest are expressed as: $T_s(P') \leqslant U_s$ (SC1), $N_n(P') \leqslant U_n$ (SC2), and $N_r(P') \leqslant U_r$ (SC3).

3.2.4 Other driver-specific constraints

When assigning drivers to shifts the skills of the drivers need to be taken into consideration. The drivers need to have competence for the rolling stock present in the shift, i.e., the path P of a driver d may only include shifts i that driver d has a proper training for (HC8). In a similar manner, the drivers can only be assigned to shifts that do not contain any work during their absences (HC9).

During pre-processing we can identify the possible shifts the individual drivers can be assigned to as we know the absences and the skills of the drivers. Thus for each driver d we can create a set $S_d \subseteq S$ that contains all the shifts i the driver d can be assigned to. Now path P of driver d may only include shifts that belong to S_d, i.e., if $i \in P(d)$ then $i \in S_d$ must be true.

3.2.5 Compactness

There are three factors that are related with the compactness of a path P. First, the rest time between two consecutive shifts, which are not separated by a double rest, should be as little over ten hours as possible. The rest time between two consecutive shifts is formulated as a cost $c^d_{i,j}$ for each edge $(i, j) \in E$

$$c^d_{i,j} = \begin{cases} s_m(j) - e_m(i) - \Delta & s_d(j) - e_d(i) \leqslant 2 \\ 0 & \text{otherwise,} \end{cases} \tag{3.11}$$

where $c^d_{i,j}$ represents the waiting time in excess to the minimum Δ between two consecutive shifts, which are not separated by a double rest. Even though the formulation (3.11) of the cost $c^d_{i,j}$ is linear, it could be nonlinear too.

Second, the number of work clusters is minimized. In order to model the compactness of a path in terms of clusters we associate a cost $c^c_{i,j}$ with each edge $(i, j) \in E$

$$c^c_{i,j} = \begin{cases} 1 & s_d(j) - e_d(i) > 2 \\ 0 & \text{otherwise.} \end{cases} \tag{3.12}$$

The cost $c^c_{i,j}$ in (3.12) is equal to one in case there exists a double week rest between the consecutive shifts i and j, otherwise it is zero.

Finally, the number of shifts that have a double week rest before and after the shift should be minimized. The goal is to avoid interrupting long rest periods by single shifts. Formally, this can be expressed as the cost

$$c^s_{i,j,k} = \begin{cases} 1 & s_d(j) - e_d(i) > 2 \ \& \ s_d(k) - e_d(j) > 2 \\ 0 & \text{otherwise,} \end{cases} \tag{3.13}$$

which holds for all edges $(i, j), (j, k) \in E$. The cost (3.13) is one, if there is a double rest between i and j, and also between j and k.

3.2.6 Objective function

The total cost of the roster R is the sum of the costs of the individual paths P that belong to R. The penalties for breaking the soft constraints for the Sunday work time, night shifts and shifts with rest for path P are computed as

$$C_s(P) = \begin{cases} T_s(P) - U_s & T_s(P) > U_s \\ 0 & \text{otherwise} \end{cases} \tag{3.14}$$

$$C_n(P) = \begin{cases} N_n(P) - U_n & N_n(P) > U_n \\ 0 & \text{otherwise} \end{cases} \tag{3.15}$$

$$C_r(P) = \begin{cases} N_r(P) - U_r & N_r(P) > U_r \\ 0 & \text{otherwise.} \end{cases} \tag{3.16}$$

Thus the total cost for breaking the soft constraints can be expressed as

$$f_s(R) = \sum_P \left(W_s C_s(P) + W_n C_n(P) + W_r C_r(P) \right), \tag{3.17}$$

where W_s, W_n, and W_r are the weights that describe the relative importance of the individual costs.

The main objective of maximizing the total artificial work hours of the regular drivers is expressed as a problem of minimizing the total artificial work hours of the extra drivers. Thereby the cost function associated with the minimization of the artificial working hours of the extra drivers in the roster R is expressed as

$$C_e(P) = \begin{cases} T_a(P) & P \text{ belongs to an extra driver in roster R} \\ 0 & \text{otherwise.} \end{cases} \tag{3.18}$$

The secondary objective of minimizing the compactness cost, $C_c(P)$, of path P is defined as the weighted sum of the costs $c^d_{i,j}, c^c_{i,j}$ and $c^s_{i,j,k}$ of the shifts traversed by the path. Let the weights for $c^d_{i,j}$, $c^c_{i,j}$, and $c^s_{i,j}$ be w^d, w^c, and w^s, respectively. Then the objectives can be expressed as $f_o(R) = \sum_P (W_c C_c(P) + W_e C_e(P))$ and the total cost of a roster is

$$f(R) = f_s(R) + f_o(R). \tag{3.19}$$

Chapter 4

Solution methodology

4.1 Background

The solution methodology is based on the algorithm by Ropke and Pisinger (2006), which is modified to take advantage of the problem-specific characteristics of the train driver rostering problem. In the Pickup and Delivery Problem with Time Windows by Ropke and Pisinger (2006), the goal is to construct routes for picking up goods and delivering them to their corresponding destination locations. The time windows for pickups and deliveries have to be satisfied, and the problem may also include capacity constraints. The authors take advantage of multiple large neighborhood insertion and removal heuristics to generate new solution candidates. The *Adaptive Large Neighborhood Search (ALNS)* algorithm tries to take the problem characteristics into consideration by evaluating the heuristics based on their past performance. The strategies that seem to offer better performance have a higher probability of being used in later iterations.

We implemented the ALNS for the train driver rostering problem due to its robustness and promising benchmark results in routing problems, which have similar properties as the rostering problem. Both problems can be viewed as network problems in which the goal is to find feasible and cost efficient paths in the network. Also, as presented in Section 2.4, heuristics are widely used within the domain of mass transit systems. Our approach assumes that the removal of a shift from a driver can never make the roster of the driver infeasible, which eliminates the possibility of having hard lower bound constraints for the number of night shifts a driver must have, for example.

4.2 Description

The solution procedure operates in three stages. In the first stage a Construction Heuristic is used to create an initial solution by assigning as many shifts as possible to the drivers. In the second stage, we try to find a feasible solution that does not contain any unassigned shifts. Finally, in the third stage the goal is to improve the solution quality by taking the objective function into consideration.

4.2.1 Initial solution

The heuristic used in the second stage requires an initial roster to operate on. This roster should be feasible in terms of the hard constraints, but it does not need to contain all the shifts of the problem. The initial roster is created using the following Construction Heuristic

1. Compute the time difference between the starting and the ending minute of each shift $i \in S$, i.e., $e_m(i) - s_m(i)$, and sort the shifts in a descending order based on this value.
2. Traverse the list from start to end. Try to insert the shift in turn to the drivers in a random order and accept the first feasible insertion. Keep list of the shifts that could not be assigned to any driver.

The motivation behind the sorting is that generally it is harder to insert the long shifts than the short shifts into the roster.

Let us call the roster produced by the Construction Heuristic as R.

4.2.2 Search for a feasible solution

We try to create a feasible roster that contains all the shifts using the *Large Neighborhood Search (LNS)* algorithm presented in Algorithm 1. The algorithm works by removing shifts from the roster and then inserting the removed shifts back to the roster in search of new, potentially better solutions, as can be seen from Figures 4.1 – 4.3.

On each iteration of the algorithm we determine the number of shifts to be removed, which can be seen on line 5. The value q is chosen randomly from a discrete uniform distribution $\mathcal{U}(q_{min}, q_{max})$, where $q_{min} \geqslant 1$ and $q_{max} \leqslant m$ are the minimum and maximum number of shifts to be removed, respectively. We always remove at least one shift and we cannot remove more than all of the m shifts in the roster R. Next we determine if we should try to insert one of the unassigned shifts to the roster together with the removed shifts. Let the probability of trying to insert one unassigned shift be p_{ins}. The heuristics used for shift removal and insertion are presented in subsections 4.2.2.1 and 4.2.2.2, respectively. If the shift insertion is successful for all the shifts, we continue to the next iteration; otherwise, we revert back to the previous solution R'. We continue this procedure until we find a roster that contains all the shifts or we reach the run time limit.

Algorithm 1 Large Neighborhood Search (LNS)

1: initial roster R
2: **repeat**
3: $\quad R' \leftarrow R$
4: $\quad$ remove $q \in \{q_{min}, ..., q_{max}\}$ shifts from roster R
5: $\quad$ insert one unassigned shift with probability p_{ins} and the q removed shifts to R
6: $\quad$ **if** we were not able to insert all the shifts to R **then**
7: $\qquad R \leftarrow R'$
8: **until** the stopping criterion is met
9: R is a feasible solution if all shifts were assigned

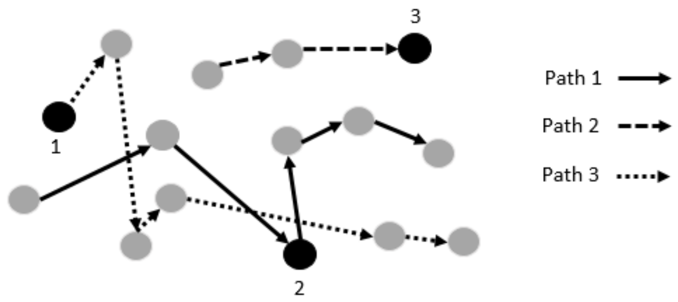

Figure 4.1: Three shifts (1-3) are chosen at random for removal from the current paths. In the example all the shifts are part of different paths.

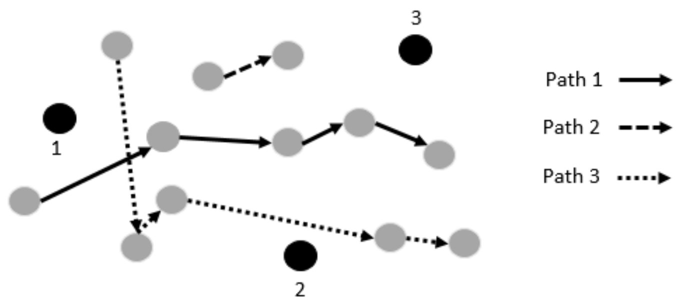

Figure 4.2: The shifts are removed and the paths are connected to produce feasible paths. The three shifts are left unassigned.

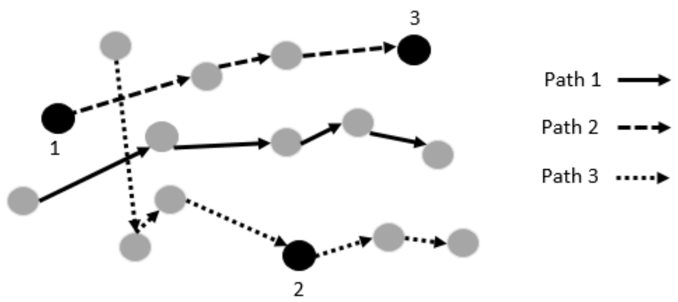

Figure 4.3: A new solution is computed by inserting the removed shifts. In the new solution, the shifts 1 and 3 are part of path 2, and shift 2 belongs to path 3.

4.2.2.1 Random removal

In random removal, we iteratively remove the required number of shifts, q, from the roster. Let us say that after i iterations we have removed $q_r < q$ shifts. Thus, the maximum number of shifts we can remove during iteration $i+1$ is $q_{i+1} = q - q_r$. Let p_1 and p_2 be parameters that have to fulfill the condition $p_1 + p_2 \leqslant 1$. Let also $N_s(d)$ denote the number of shifts assigned to driver d. The removal procedure for iteration $i+1$ is described in Algorithm 2. The procedure is repeated until q shifts have been removed.

Algorithm 2 Random removal iteration

```
roster R
choose randomly a driver d from the roster R
choose a random number r from the uniform distribution U(0,1)
if N_s(d) = 0 then
    no shifts can be removed from the driver
else if (N_s(d) = 1) or (q_{i+1} = 1) or (r ⩽ p_1) then
    remove one shift randomly from driver d
else if (N_s(d) = 2) or (q_{i+1} = 2) or (r ⩽ p_1 + p_2) then
    remove randomly two consecutive shifts from d
else
    remove randomly three consecutive shifts from d
```

The motivation for the emphasis of removal of consecutive shifts comes from practical experience. Usually the rosters require changing larger portions of the paths of the individual drivers in order to find new solutions, which we can emphasize by tuning the parameters p_1 and p_2.

4.2.2.2 Random insertion

Random insertion is a simple strategy to insert shifts to a roster. We try to insert each of the removed shift to the drivers in a random order. We accept the first driver we can assign the shift to. If we can assign all the shifts to the drivers the insertion has been successful.

4.2.3 Solution improvement

In the previous stage, the goal was to produce a feasible roster. The objective in the third and final stage is to maintain feasibility, while taking the costs of the individual paths (see Section 3.2.6) into consideration in order to minimize the overall cost of the roster. This is done using an adaptive version of the Large Neighborhood Search. In the adaptive version, we utilize four insertion strategies instead of one, namely greedy random insertion, greedy insertion, regret-2 insertion, and regret-3 insertion. These strategies are described in more detail in Sections 4.2.3.1 – 4.2.3.3.

The Adaptive Large Neighborhood Search, which is presented in Algorithm 3, reuses most of the ideas from the LNS algorithm introduced in Algorithm 1. At each iteration we remove q shifts using the random removal strategy, but the insertion strategy is chosen based on the past performance of the strategies. Inferior, but feasible solutions, may be accepted as solutions with probability p_{acc}. The probability depends on the relationship between the new solution R and the old solution R', i.e., $p_{acc} = p_{acc}(f(R), f(R'))$. A higher probability p_{acc} reduces the chances that the algorithm cannot escape a local optimum, but it can also slow the convergence down.

Algorithm 3 Adaptive Large Neighborhood Search (ALNS)

feasible roster R
$R_{best} \leftarrow R$
repeat
    $R' \leftarrow R$
    choose a random number r from the uniform distribution $\mathcal{U}(0, 1)$
    remove $q \in \{q_{min}, ..., q_{max}\}$ shifts from roster R
    choose insertion strategy s
    insert the q removed shifts back to R using strategy s
    if insertion did not result in a feasible roster **then**
        $R \leftarrow R'$
    else if $(f(R) > f(R'))$ and $(p_{acc}(f(R), f(R')) < r)$ **then**
        $R \leftarrow R'$
    else if $f(R) < f(R_{best})$ **then**
        $R_{best} \leftarrow R$
until the stopping criterion is met
R_{best} contains the best feasible solution found

4.2.3.1 Greedy random insertion

The q shifts to be inserted to the roster are assigned sequentially in random order to the drivers. Let $f_{i,d}$ be the change in the objective function value when shift i is assigned to $d \in \{1, ..., n\}$, where n denotes the number of drivers. If the insertion is infeasible, then let $f_{i,d} = \infty$. Using this metric, we assign i to the driver that corresponds to the value $d_i = \arg\min_d \left\{f_{i,d}\right\}$. If we cannot assign i to any of the drivers then the insertion has failed and we terminate.

4.2.3.2 Greedy insertion

The greedy insertion assigns at each step the shift that corresponds to the best change in objective function value to the roster R. Let the set $L \subseteq Q$ be a collection of the removed shifts that have not yet been reinserted. The change in the objective function value when $i \in L$ is assigned to the corresponding best driver $d_i \in \{1, ..., n\}$ is $f_i = \min_d \left\{f_{i,d}\right\}$. If $f_i = \infty$ for any i we terminate, otherwise we continue. Out of all the shifts $i_{min} = \arg\min_i \{f_i\}$ has the overall lowest insertion cost. Thus i_{min} is assigned to driver $d_{i_{min}}$. If $|L| \geqslant 1$, this insertion process is continued using the set $L' = L \setminus \{i_{min}\}$.

4.2.3.3 Regret heuristic for insertion

In the k-regret we try to compute value that measures how much it is worth to assign a shift to the driver that corresponds to the best change in objective function value. The procedure works in the following manner. We compute the change in the objective function value of assigning the shift i to each of the drivers $d \in \{1, ..., n\}$. This is done for all the shifts which have not yet been assigned. Let the smallest objective function value corresponding to i be f_i^1, the second smallest f_i^2, and the k:th smallest f_i^k. Also let d_i be the driver that corresponds to the smallest cost of i, f_i^1. If $f_i^1 = \infty$ for any i we terminate, otherwise we continue. In the general k-regret heuristic with $k \in \{1, ..., n\}$, the shift i that maximizes $\sum_{j=1}^{k} \left(\Delta f_i^j - \Delta f_i^1\right)$ is assigned to d_i. This process is continued until all the shifts have been assigned, or until one of the still unassigned shifts cannot be assigned to any driver. In the latter case the insertion has failed.

The parameter k determines the complexity of the selection criteria: the larger k is, the more we look ahead in the insertion process and try to estimate the best way to assign all the shifts, not only the individual shifts. In the ALNS we use the two most simple strategies: 2-regret and 3-regret heuristics.

Chapter 5

Results

5.1 Pre-processing

Pre-processing involves capturing the essential characteristics of the shifts from the raw data and formulating these characteristics into a format that can be easily read by the optimization algorithm. The raw data is gathered from the planning software in the form of a text file. This text file is exported separately for each planning period, such as the shifts are subject to changes depending on driver feedback, transportation needs and driver availability. The export contains the shift data for all the depots. The data for a specific depot can easily be filtered using the corresponding depot id.

The data for an individual depot contains all the shifts of the depot and all the tasks the shifts consist of. Based on the information of the individual tasks, we can compute the necessary characteristics of the shifts. For example, the starting time of the shift is determined by the starting time of the earliest task that belongs to the shift. In a similar manner, the ending time of the shift is the ending time of the last task. The approach of building shift attributes from task level information makes it possible to associate robustly new, possibly complex, attributes with the shifts.

Various driver-specific data needs to be collected before the optimization algorithm can be called. Most of the data can be computed from the rosters of the previous work period based on the ending time of the drivers' last shift. The drivers' current training status and their future absences can be gathered from the company's internal human resource software. Based on the data about driver absences we can compute the artificial work hour limits of the drivers and the possible shifts the drivers can be assigned to.

5.2 Test cases

The solution methodology is implemented in C++. The executable file of the algorithm loads in the data processed by R, as described in Section 5.1, and creates a text file that contains the solution. The solution can then be visualized using an R function implemented for creating simple and readily understandable figures of the rosters.

Three different problem cases are used to evaluate the performance of algorithm. The test cases, which are presented in Table 5.1, consist of real world shift data from three different depots in Finland. The smallest problem contains 111 shifts compared to the 747 shifts in the largest problem. There are also differences in the other attributes of the test cases, for example in the amount of night work. Having benchmark problems with different characteristics gives a clear picture of the overall performance of the optimization algorithm. The test problems are solved using a computer with Intel Core i5-6200U processor and 8GB of RAM. The maximum frequency of the processor is 2.80 GHz.

Table 5.1: The problem characteristics of the three different test cases. The problems differ in size as well as in other attributes, such as night work.

Attribute	small	medium	large
Shifts	111	217	747
Night shifts	43	129	413
Shifts with rest	35	131	119
Total artificial work hours	1155	2797	7516
Night work hours	148	613	2425
Sunday work hours	189	510	889

The solution quality is evaluated using the following statistics. First, we check how close the average artificial work hours of the regular drivers are to the upper limit. We also check how well we can limit the number of night shifts, shifts with rest, and Sunday work hours of the individual drivers. We measure the compactness roster using the number of days off at the double week rests.

For each test problem, the total computation time was set to 60 seconds. No other convergence criteria were used apart from the time limit. The time limit was set low to determine how good solutions the algorithm is able to find in short time spans.

5.2.1 Small problem

The small optimization problem was solved using eleven drivers, out of which two were extra drivers. The final results are in Table 5.2. As the total amount of work assigned to the extra drivers is minimized, all the result attributes presented in the table are only for the regular drivers.

Table 5.2: Result statistics for the small rostering problem solution.

Attribute	min	max	mean	upper limit
Shifts	10	12	10.89	
Night shifts	3	5	4.11	5
Shifts with rest	2	4	3.56	4
Artificial work hours	113.5	114.45	114.44	114.75
Night work hours	12.78	18.18	14.53	42
Sunday work hours	13.35	24.75	18.72	25
Total days off at double week rests	6	8	7.11	

To balance the strenuous shifts, the maximum number of night shifts and shifts with rest was set to five and four, correspondingly. The upper bound for the Sunday work hours was set to 25. The limits for the artificial work hours and for the night work hours are those defined by the union contracts. The model does not include any limits for the maximum number of shifts. The lower limit for the total consecutive days off at the week rests is six in a three week work period, because the maximum number of consecutive work days is five.

A part of the roster created by the algorithm can be seen from Figure 5.1. The green boxes represent the shifts, the numbers on the columns are the day numbers and the rows correspond to the drivers. Starting from the top of each shift we have the starting time of the shift, the id of the shift, and, finally, the ending time of the shift. The shifts with the light green colour are day shifts, the bright green shift is a night shift of type A, and the dark green shifts are night shifts of type B. The day numbers with grey background represent weekends.

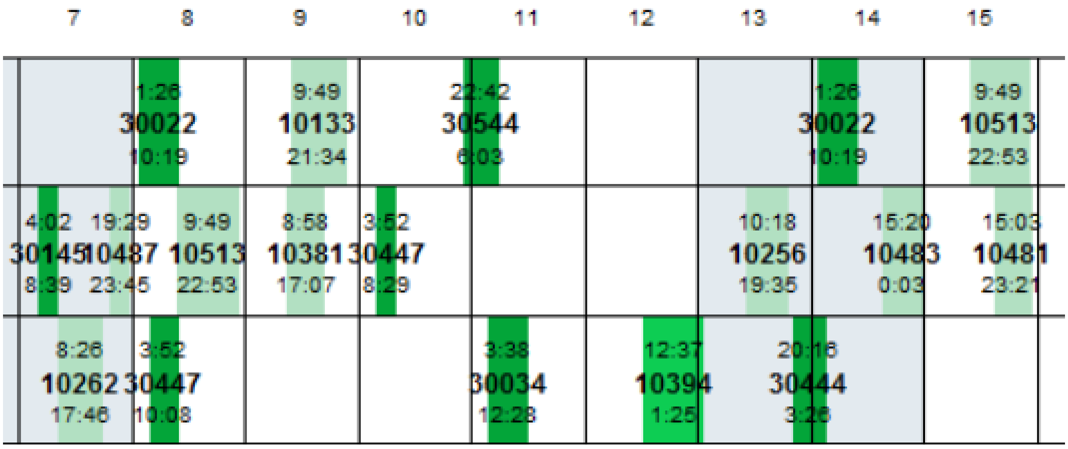

Figure 5.1: A snippet of the roster created for the small test problem.

The result looks excellent in that the mean artificial work hours are only eighteen minutes from the upper limit and the strenuous shifts are well distributed among the drivers. On average, the drivers also have a three day week rest during the three week long work period.

5.2.2 Medium sized problem

The medium sized problem is a step up from the small problem with an increase of almost 100% in the number of shifts and an increase of over 250% in the total artificial work hours. Thus, solving the problem requires more drivers. The final solution contains 23 regular drivers and 6 extra drivers.

As in the small problem, the results for the medium sized problem are good. The average artificial work hours are not as close to the upper limit as for the small problem, which can be seen from the Table 5.3. The mean number of night shifts and shifts with rest are close to the upper limits, which signals that the upper limits are meaningful and actually constrain the solution space. The total consecutive days off at week rest is 0.65 calendar days over the six calendar days required by the union contract.

Table 5.3: Result statistics for the medium sized rostering problem solution.

Attribute	min	max	mean	upper limit
Shifts	7	10	8.61	
Night shifts	3	6	5.00	6
Shifts with rest	3	6	5.48	6
Artificial work hours	110.20	114.70	113.51	114.75
Night work hours	11.28	35.02	23.35	42
Sunday work hours	14.80	24.80	21.93	25
Total days off at double week rests	6	9	6.65	

Next we tried changing the upper limit for the number of shifts with rest from six to seven to see if that would increase the mean artificial work hours. We did see a relatively small increase of ten minutes in the mean artificial work hours, which suggests that the upper bound for the number of shifts with rest is not the only factor that constrains the problem tightly.

5.2.3 Large problem

The largest problem contains a large amount of night work, which makes the problem significantly more difficult, because there are multiple constraints concerning night shifts and night work overall. The problem solution uses 71 drivers, out of which nine were extra drivers. The solution statistics are presented in Table 5.4.

The artificial work hours are only a few minutes from the upper limit, even though the average of night work hours is over 36 hours, while the upper limit is 42 hours. The mean Sunday work hours are close to the upper limit, which is good. The same remark holds also for the night shifts. Because the number of shifts with rest is quite low, the need for balancing the shifts with rests is not of high importance.

Table 5.4: Result statistics for the large rostering problem solution.

Attribute	min	max	mean	upper limit
Shifts	9	13	11.4	
Night shifts	4	7	6.2	7
Shifts with rest	0	3	1.67	3
Artificial work hours	114.42	114.75	114.66	114.75
Night work hours	18.78	41.58	36.30	42
Sunday work hours	9.12	14.95	13.51	15
Total days off at double week rests	6	11	7.08	

Chapter 6

Practical experiences

The optimization model, which has been in use since summer 2016 for a major part of the depots, has been successful in that it has increased the utilization rate of the drivers by $3-5\%$. The results produced by the model are excellent, especially because it considers several aspects of work well-being already at the outset. Using the model the planning department has been able to allocate the night work, Sunday hours and shifts with rest more evenly among the drivers. The planners experience that they are able to efficiently create rosters that are considered less strenuous by adjusting the constraints, such as the minimum rest time between two consecutive shifts or the maximum work hours in a work cluster.

The planners have on multiple occasions created and presented alternative solutions to the union representatives based on the discussions between the planners and the representatives. The solutions have been equally good from the employer's point of view, and thus the representatives have been able to freely choose the solutions they prefer. However there have been comments about the choices being too similar even though different properties have been emphasized, which usually occurs due to the fact that a high importance is given to maximizing the artificial work hours.

The time required to plan the rosters has reduced by several hours per depot. When using the optimization model a large share of the planners' time is occupied by collecting the input data for the model. However, once the data has been collected, new solutions can be computed in a few minutes and unexpected changes to the input data can thus be readily taken into consideration comfortably. Using the model the planners have considerably more time to focus on future improvements and development ideas regarding the rostering process, because the manual rostering does not occupy such a large share of the planners' time.

From a strategic perspective, the model has allowed the planners to experiment whether certain constraints proposed by the union representatives should be implemented in practice. There have been suggestions for example to ban certain shift combinations, such as having a night shift and a shift with rest on consecutive calendar days. Based on the results of the experiments, the planners and the union representatives have assessed whether the constraints should be included in the model and how the constraints would affect the overall solution quality.

Apart from rostering, the model has also been used to simulate how changes to the shifts affect the rosters. Based on the simulation results the planners have been able to adjust the shifts with the hopes of improving the rosters. The algorithm can also be used to plan future recruitments for a depot, as it can be used to determine the minimum number of drivers needed to take care of the workload of the depot.

Much feedback has been collected concerning the automatized solution. The planners have experienced that the solutions provided by the algorithm are difficult to modify manually without worsening the overall solution quality significantly. The planners usually have to do manual changes to the solutions after getting feedback from the union representatives. The future goal is minimize the need for these manual changes by focusing more on the major guidelines regarding the work well-being of the drivers.

The planning department has adopted the new planning approach well. The planners have been able to collect input data and run results without the need of technical support. The planners have also provided essential feedback on methods to improve the process further, especially the way the input data is collected and stored. Based on the feedback, the process has already been improved during the writing of this thesis.

Chapter 7

Conclusions

7.1 Summary

The primary goal of this project was to increase the utilization rate of the regular drivers, thus minimizing the amount of work left for the extra drivers. This goal was met with considerable success as the average artificial work hours of the drivers have increased clearly compared to the manually planned rosters, while the work load of the extra drivers has reduced.

Increasing the work load of the regular drivers created a need to ensure that the work well-being of the drivers is well taken care of. The algorithm has enabled the planners to analyze and fix issues related to work well-being by adjusting model constraints in a suitable manner.

A third goal of the thesis was to improve the efficiency of the planning process by reducing the amount of sheer manual work. Based on feedback from the planners, the rostering tool has sped up the planning process, while increasing their capability to tackle sudden changes that affect the rosters. By reducing the time spent on manual rostering the tool has also enabled the planners to focus more on other tasks related to resource planning.

7.2 Future research

The development of the rostering tool is an ongoing process that will see changes in the three different aspects of the tool: problem model, optimization algorithm, and integration to existing software and databases. The model needs to be modified if the underlying constraints that are based on legislature or union contracts change, or if new constraints related to the work well-being of the drivers are added. There are plans to include multiple driver-specific constraints, other than the artificial work time limit, so that the planners could use the model to create profiles for each depot, or even for the individual drivers. For example, certain drivers prefer having long rests between two consecutive shifts instead of having over two rest days at the double week rests, while others prefer the opposite. There are also drivers who prefer starting work early in the morning, while others have no issues working late into the night. Including driver-specific constraints to the model is not an issue, however the constraints have to be evaluated thoroughly to find out if they produce results that are of high quality both from the drivers' and the employer's point of view. In the future new attributes will also be associated with the shifts, for example how well received a shift is, with the goal of creating a versatile and thorough description of the shifts. These new attributes are then generally distributed evenly among the drivers to increase the fairness of the roster.

Based on the complexity of the underlying problem model, certain performance improvements may need to be implemented to the optimization algorithm. The current performance of the model is excellent, especially when taking into consideration that the model is ran on typical business laptops. Thus the performance could easily be improved simply by running it on better performing hardware, for example on a server. It would also be possible to parallelize multiple functions of the algorithm, e.g., the feasibility check for the drivers, because the feasibility of a single driver's roster does not depend on any other driver. The need for performance improvements needs to be analyzed case by case. For example, performance issues originating from badly formulated constraints may not require any changes to the actual algorithm, while creating a high performing version specifically for reducing the memory footprint would make it necessary to modify the nature of the algorithm.

There are ideas to create an integrated process around the algorithm that would manage all tasks related to data input and output. Currently collecting the required data involves multiple manual tasks, which is not ideal, because these tasks could be automatized for the most part. Also the rosters produced by the algorithm have to be inserted manually to the planning software, which is an issue left to be solved.

Bibliography

S. Abdennadher and H. Schlenker. Interdip–an interactive constraint based nurse scheduler. In *Proceedings of The First International Conference and Exhibition on The Practical Application of Constraint Technologies and Logic Programming, PACLP*, 1999.

U. Aickelin and K. A. Dowsland. Exploiting problem structure in a genetic algorithm approach to a nurse rostering problem. *Journal of Scheduling*, 3(3):139–153, 2000.

F. Bellanti, G. Carello, F. Della Croce, and R. Tadei. A greedy-based neighborhood search approach to a nurse rostering problem. *European Journal of Operational Research*, 153(1):28–40, 2004.

L. Bianco, M. Bielli, A. Mingozzi, S. Ricciardelli, and M. Spadoni. A heuristic procedure for the crew rostering problem. *European Journal of Operational Research*, 58(2):272–283, 1992.

R. Borndörfer, C. Schulz, S. Seidl, and S. Weider. Integration of duty scheduling and rostering to increase driver satisfaction. *Public Transport*, 9(1):177–191, 2017.

E. K. Burke, P. De Causmaecker, G. V. Berghe, and H. Van Landeghem. The state of the art of nurse rostering. *Journal of Scheduling*, 7(6):441–499, 2004.

K. W. Campbell, R. B. Durfee, and G. S. Hines. Fedex generates bid lines using simulated annealing. *Interfaces*, 27(2):1–16, 1997.

P. Cappanera and G. Gallo. On the airline crew rostering problem. Technical Report TR-01-08, Department of Computer Science, University of Pisa, Italy, 2001.

A. Caprara, M. Fischetti, P. Toth, D. Vigo, and P. L. Guida. Algorithms for railway crew management. *Mathematical Programming*, 79(1-3):125–141, 1997.

A. Caprara, P. Toth, D. Vigo, and M. Fischetti. Modeling and solving the crew rostering problem. *Operations Research*, 46(6):820–830, 1998.

B. Cheang, H. Li, A. Lim, and B. Rodrigues. Nurse rostering problems - a bibliographic survey. *European Journal of Operational Research*, 151(3): 447–460, 2003.

I. T. Christou, A. Zakarian, J.-M. Liu, and H. Carter. A two-phase genetic algorithm for large-scale bidline-generation problems at delta air lines. *Interfaces*, 29(5):51–65, 1999.

H. Dawid, J. König, and C. Strauss. An enhanced rostering model for airline crews. *Computers & Operations Research*, 28(7):671–688, 2001.

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

B. Gopalakrishnan and E. L. Johnson. Airline crew scheduling: state-of-the-art. *Annals of Operations Research*, 140(1):305–337, 2005.

T. Grossman, D. Samuelson, S. Oh, and T. Rohleder. Call centers. Technical report, Haskayne School of Business, University of Calgary, 1999.

M. E. Horn, H. Jiang, and P. Kilby. Scheduling patrol boats and crews for the Royal Australian Navy. *Journal of the Operational Research Society*, 58 (10):1284–1293, 2007.

A. Ikegami and A. Niwa. A subproblem-centric model and approach to the nurse scheduling problem. *Mathematical Programming*, 97(3):517–541, 2003.

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

J. König and C. Strauss. Rostering-integrated services and crew efficiency. *Information Technology & Tourism*, 3(1):27–39, 2000a.

J. König and C. Strauss. Supplements in airline cabin service. In D. R. Fesenmaier, S. Klein, and D. Buhalis, editors, *Information and Communication Technologies in Tourism 2000*, pages 365–374. Springer Vienna, 2000b.

H. Li and K. Womer. A decomposition approach for shipboard manpower scheduling. Technical report, University of Missouri, St Louis, 2009.

V. Mehrotra. Ringing up big business. *OR/MS Today*, 24:18–25, 1997.

H. H. Millar and M. Kiragu. Cyclic and non-cyclic scheduling of 12 h shift nurses by network programming. *European Journal of Operational Research*, 104(3):582–592, 1998.

X. Qi and J. F. Bard. Generating labor requirements and rosters for mail handlers using simulation and optimization. *Computers & Operations Research*, 33(9):2645–2666, 2006.

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

S. Ropke and D. Pisinger. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science*, 40(4):455–472, 2006.

M. Sabar, B. Montreuil, and J.-M. Frayret. A multi-agent-based approach for personnel scheduling in assembly centers. *Engineering Applications of Artificial Intelligence*, 22(7):1080–1088, 2009.

R. Silvestro and C. Silvestro. An evaluation of nurse rostering practices in the national health service. *Journal of Advanced Nursing*, 32(3):525–535, 2000.

R. Stolletz. Operational workforce planning for check-in counters at airports. *Transportation Research Part E: Logistics and Transportation Review*, 46(3):414–425, 2010.

L. Talarico and P. A. M. Duque. An optimization algorithm for the workforce management in a retail chain. *Computers & Industrial Engineering*, 82:65–77, 2015.

V. Valdes and V. Andres. *Integrating Crew Scheduling and Rostering Problems*. PhD thesis, University of Bologna, 2010.

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

D. M. Warner. Scheduling nursing personnel according to nursing preference: A mathematical programming approach. *Operations Research*, 24 (5):842–856, 1976.

L. Xie and L. Suhl. Cyclic and non-cyclic crew rostering problems in public bus transit. *OR Spectrum*, 37(1):99–136, 2015.

T. H. Yunes, A. V. Moura, and C. C. De Souza. Hybrid column generation approaches for urban transit crew management problems. *Transportation Science*, 39(2):273–288, 2005.