Master thesis

Work Distribution of a Heterogeneous Library Staff - A Personnel Task Scheduling Problem

Claes Arvidson, Emelie Karlsson

LiTH - MAT - EX - - 04 / 04 - - SE

Work Distribution of a Heterogeneous Library Staff - A Personnel Task Scheduling Problem

Optimeringslära, Linköpings Universitet

Claes Arvidson, Emelie Karlsson

LiTH - MAT - EX - - 04 / 04 - - SE

Exam work: **30 hp**

Level: **A**

Supervisor: **T. Larsson**,
Optimeringslära, Linköpings Universitet

Examiner: **E. Rönnberg**,
Optimeringslära, Linköpings Universitet

Linköping: **June 2016**

Abstract

Here is where you can write your abstract. It may be very long, or it may be very short, the reason you have an abstract is for people not to be forced to read lots of crap.

But still, they will have to read your abstract. After all, the abstract is what everyone reads...

Keywords: Keyword One, Chemostat, Another Key-Word, Key, Clé, Mot de cle, Nyckelhål, XBOX, Dagens viktigaste nyckelord, and Keywords.

URL for electronic version:

http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-77777

Acknowledgements

I would like to thank my supervisor, I would like to thank my supervisor, I would like to thank my supervisor, I would like to thank my supervisor...

I also have to thank, I would like to thank my supervisor, I would like to thank my supervisor, I would like to thank my supervisor, I would like to thank my supervisor...

My opponent NN also deserves my thanks, I would like to thank my supervisor, I would like to thank my supervisor, I would like to thank my supervisor...

Nomenclature

Most of the reoccurring definitions, symbols and abbreviations are described here.

Definitions

Plocklista	Text
Library on wheels	Text
Rostering	Text
Matheuristic	Text
Initial schedule (?) Basic schedule (?)	

Symbols

Abbreviations

Exp	Text
Info	Text
PL	Text
PTSP	Text
SMPTSP	Text

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

At a library, staff absence can cause problems and result in a shortage of staff with a special competence. If a worker is unavailable a certain day due to a meeting or illness it would require a stand-in to fill the vacancy. Therefore, it is of great interest to create schedules with as many skilled stand-ins as possible to overcome such disturbances, when they occur. Furthermore, the library personnel have certain demands and preferences as to how a satisfactory working schedule should be. For instance, it is neither preferable to work more than one evening per week nor work more weekends than required.

The problem addressed in this thesis work concerns the library staff at the Central Library of Norrköping. The library has more than 40 employees and the culturally important building from the 1950's is known to all inhabitants of Norrköping. The library is open weekdays between 08:00 and 20:00 and during weekends between 11:00 and 16:00. The generous opening times also creates a challenge for the library as it requires a large pool of well coordinated personnel to keep it running. In addition, the library also provides its services to one other smaller library. This creates a challenge for the library schedulers.

1.2 Problem description

In this section, the most important features of the worker scheduling problem studied in this thesis work will be explained.

1.2.1 Description of the daily tasks at the library

The most important activity at a library is the activity directed towards the public. This includes book loan services as well as providing customers with helpful information about the resources at the disposal of the library. These activities are referred to as "outer tasks". In addition, the book collection must be maintained, the returned books must be sorted and put back to the shelves, the web page must be updated and so on. Such work is often referred to as "inner work" and this non-public work is an equally important part of the everyday activities at the library.

Three main outer task types can be identified at the library of Norrköping, working in the service counter (sv. expiditionsdisken), working in the information counter (sv. informationsdiken) and assembling books which are to be sent to other libraries according to the "fetch list" (sv. plocklista). The fetch list is a task type for which the worker is scheduled during a whole day, while the other task types have two to five hour shifts. The outer tasks can be performed by either librarians or both, as described in Table 1.1.

Table 1.1: Outer tasks can be performed exclusively by librarians or by both librarians and assistants.

Task	**Description**	**Qualification**
Service counter (Exp)	Administring loans, library cards and the loaning machine	Librarian or assistant
Information counter (Info)	Handling questions about the library's resources.	Librarian
Fetch list (PL)	Fetching books that are to be sent to other libraries.	Librarian or assistant
Hageby (HB)	Handling librarian tasks at the filial Hageby during weekends.	Librarian
Library on Wheels (BokB)	Driving the Library on Wheels to different areas of town.	Librarian

Since the number of visitors at the library varies throughout the day and also during different days of the week, so does the demand for personnel for the three tasks. The demand of personnel for the different task types is illustrated in Table 1.2, according to figures given by the library.

Table 1.2: Demand of staff for the three daily tasks.

Day	**Time**	**Exp demand**	**Info demand**	**PL demand**[1]
Mon-Fri	08:00-10:00	2	2	1
Mon-Fri	10:00-13:00	3	3	1
Mon-Fri	13:00-16:00	3	3	1
Mon-Fri	16:00-20:00	3	3	-
Sat	11:00-16:00	3	3	-
Sun	11:00-16:00	3	3	-

As in the case with most libraries, the Central Library of Norrköping also has responsibilities that fall outside of its normal daily activities. One such responsibility is the running of a smaller library filial in Hageby, situated in a suburban area of Norrköping, during weekends. It is decided that only librarians are qualified for this task, as it implies some tasks that need certain knowledge.

Table 1.3 shows the demand of staff at Hageby. Worth noting is that it is the same person working at Hageby a weekend due to a couple of reasons. Since a worker is supposed to work both Saturday and Sunday when due for a weekend, it is more desirable to let the worker focus on the same task both days. Furthermore, since Hageby is located in a suburban, some travel distance is required to reach it. This is compensated by letting the worker at Hageby be free from work Friday evening, which otherwise is included in the weekend work.

Table 1.3: Demand of staff at Hageby

Day	Time	HB demand
Sat	11:00-16:00	1
Sun	11:00-16:00	1

Similarly, only a handful librarians are qualified for the task known as the "Library on Wheels" (sv. Bokbussen), which is a library bus that provides citizens in remoter areas of the city with books and other library services. The Library on Wheels only operates a few times a week and the schedule differs between odd and even weeks.

Table 1.4: Demand of staff Library on Wheels

Odd Week	Mon	Tue	Wed	Thu	Fri
08:00-10:00	1	0	1	1	1
16:00-20:00	1	0	1	1	0
Even Week	**Mon**	**Tue**	**Wed**	**Thu**	**Fri**
08:00-10:00	1	0	1	1	0
16:00-20:00	0	0	1	1	0

Apart from the outer work described above, there is also inner work at the library which sometimes need to be scheduled. One such type of inner work is meetings, as it concerns a large number of staff. Both library meetings, scheduled for the whole work force, and department meetings, scheduled only a subset of workers, exist. The library meetings are fixed in time and take place every fifth weeks on Monday mornings 8:15-10:00 while the department meetings are more flexible and take place every fifth week. For this, all department members must be available. Only three departments have scheduled meetings: the child department, the adult department and the media department.

1.2.2 Personnel attributes

At a library, the main group of workers are librarians and assistants (also janitors, cleaners, security guards etc.). The librarians make sure the library's resource collection is up to date, that the visitors find what they're looking for and perform a larger number of inner tasks. Assistants can perform most of theses tasks, while others require librarian competence.

The Central Library of Norrköping currently has 39 workers, 23 of which are librarians and 16 of which are assistants. All staff have different availability for performing tasks, depending on their working hours and the amount of inner work they are in charge of. In the standard case, each worker is assigned one evening per week and once per five weeks he/she is assigned to work during the weekend. The following week after a weekend is compensated by two free days, placed according to the wishes of the worker.

Let us consider a sample worker who is qualified for the same tasks as a librarian, works full time and is also assigned to work on Wednesday evenings. The worker is also assigned to work weekend on the fourth week and has chosen to take out its days off on Thursdays and Fridays on week five. The workers availability is thus illustrated in table 1.5. The schedule repeats itself after

five weeks and illustrates only the availability for tasks but says nothing about whether the worker has been assigned any tasks or not.

All workers have a five week schedule in the same manner as the sample worker. However, in order to meet the weekend demands illustrated by Tables 1.2, 1.4 and 1.3, it is evident that not all workers can be assigned for weekend work at week 4. Instead, a worker's schedule should be seen as a *relative schedule*, which can be shifted by whole weeks. We refer to this as the *week rotation*.

The relative schedule is relative to the *overall schedule* for the library. If the sample worker for example has week rotation 1, this means the weekend week in his relative schedule would be shifted to week 1 in the overall schedule. In Table 1.5, the weekend rotation is 4.

Table 1.5: Availability schedule for a sample worker. Yellow signifies that the worker is available. In parenthesis, the weekend shift. The week rotation is 4

Week 1	**Mon**	**Tue**	**Wed**	**Thu**	**Fri**	**Sat**	**Sun**
08:00-10:00 (11:00-16:00)							
10:00-13:00							
13:00-16:00							
16:00-20:00							
Week 2	**Mon**	**Tue**	**Wed**	**Thu**	**Fri**	**Sat**	**Sun**
08:00-10:00 (11:00-16:00)							
10:00-13:00							
13:00-16:00							
16:00-20:00							
Week 3	**Mon**	**Tue**	**Wed**	**Thu**	**Fri**	**Sat**	**Sun**
08:00-10:00 (11:00-16:00)							
10:00-13:00							
13:00-16:00							
16:00-20:00							
Week 4	**Mon**	**Tue**	**Wed**	**Thu**	**Fri**	**Sat**	**Sun**
08:00-10:00 (11:00-16:00)							
10:00-13:00							
13:00-16:00							
16:00-20:00							
Week 5	**Mon**	**Tue**	**Wed**	**Thu**	**Fri**	**Sat**	**Sun**
08:00-10:00 (11:00-16:00)							
10:00-13:00							
13:00-16:00							
16:00-20:00							

Considering again the sample worker's schedule, it may look as if the worker can be scheduled for tasks at any yellow shift. However, there are regulations set by the library controlling how the outer tasks are to be distributed among the workers. The purpose of the regulations is to guarantee that the worker schedules are not too unbalanced or uncomfortable.

There four basic demands of the resulting schedule. Firstly, workers are only allowed to take at a maximum one task per day. Secondly, workers only work

at most weekend per five weeks. During this weekend it is customary that you work Friday evening shift, Saturday and Sunday shift consecutively, unless you are scheduled for Hageby, which is far away and is thus compensated by not entailing Friday evening work. Thirdly, weekend work is to be compensated by days off. There are a few possible variations of the weekend compensation, but usually the worker takes one and a half or two days off the week following upon weekend work. Lastly, a worker is only allowed to work one evening per week, excluding the Friday which belongs to the weekend week.

An example of a feasible schedule for the sample worker is provided in Table 1.6. It should be noted that this schedule is created for a general worker and does not necessarily apply to all workers. Among workers, special availability restrictions due to odd-and even weeks exist and only a subset of the librarians are available for the Library on Wheels and Hageby. Also, some workers never work evening or weekend shifts.

Table 1.6: Example of a feasible first week for the sample worker.

Week 1	**Mon**	**Tue**	**Wed**	**Thu**	**Fri**	**Sat**	**Sun**
08:00-10:00 (11:00-16:00)		Exp		PL			
10:00-13:00	Exp			PL			
13:00-16:00				PL	Info		
16:00-20:00			Info				

1.2.3 Scheduling objectives: stand-in maximization and schedule variation

What is investigated in this thesis is not primarily how to distribute the tasks according to the rules given above, but rather how to do this is a way so that the number of stand-in personnel is maximized. The emergency back-up system of stand-ins is crucial for the library in order to be able to keep the library desks open and fully staffed. Thus, the highest priority is to maximize the number of stand-in personnel.

A stand-in is defined as a worker who is available for outer tasks during the first three shifts, as well as not scheduled for any shift that day. Inner work sometimes prevents personnel from being scheduled as a stand-in for the outer tasks, and therefore also information about such work must be known to the scheduler. Both librarians and assistants can be scheduled as a stand-in, but only librarians are qualified to take emergency shifts at the Info desk. Since the regular library activity is the most crucial activity, there is no need for stand-ins during evening and weekends. Similarly, there are no assigned stand-ins for the Library on Wheels or for Hageby.

Apart from maximizing the number of stand-ins for each day at the library, another important measurement of a good schedule is the level of variation. It is desirable for the sake of the personnel to create schedules in which the weeks resemble eachother or are repeated according to some pattern. On the other hand, since some shifts are more attractive than other and in order to avoid too much repetitiveness it is desirable that the days during a work week are not too similar. The example schedule in Table 1.6 is an example of a sufficiently varied weekly schedule.

1.3 Method

1.4 Topics Covered

Chapter 2

Literature review

The scheduling problem is a mathematical optimization problem which has been studied since the 1950s and concerns creating a feasible and satisfactory schedule for workers or machines performing tasks. One of the most extensive overviews in the area is provided by Ernst et al. (2004). They state that, although the complexity of the scheduling problem has not increased in recent years, the mathematical models used to solve the scheduling problems have become more realistic and refined. Modern scheduling problems often concern the distribution of tasks and the creation of worker shifts, also taking softer values into account, such as worker satisfaction and worker fatigue. Due to this modeling refinement as well as the development of more powerful computational methods, it is possible today to solve scheduling problems in a more satisfactory way than before.

In this chapter, the scheduling problem is classified into different subcategories which are areas related to the thesis work presented in this paper. The relevant areas for our work include Personnel Task Scheduling Problems (PTSP), Shift Minimization Task Scheduling Problems (SMTSP), Tour Scheduling Problems (TSP) and a few variations of these. Also problems featuring soft constraints will be studied in an additional section. Models and solution methods of these problems will be discussed under each headline and a summary will be provided at the end together with a discussion about the relevance of the studied articles to our thesis work.

2.1 Personnel Task Scheduling Problem

In many real life situations production managers will face the Personnel Task Scheduling Problem (PTSP) while scheduling service operations. Krishnamoorthy and Ernst (2001) writes in their article that the PTSP occurs when the rosterer or shift supervisor need to allocate tasks with specified start and end times to available personnel who have the required qualifications. It also occurs in situations where tasks of fixed times shall be assigned to machines. Decisions will then have to be made regarding the amount of maintenance workers needed and which machine the workers are assigned to look after.

There are numerous variants to the PTSP. Studies on these have been made by Krishnamoorthy and Ernst (2001) who gives a list of attributes that com-

monly appear in a PTSP, which are listed in Table 2.1. Furthermore, there are traits that always appear in a PTSP; tasks with fixed start and end time are to be distributed to staff members that possess certain skills, allowing them to perform only a subset of the available tasks. Start and end time of their shifts are also predetermined for each day.

One variant, which also is the most simple, is mentioned in Krishnamoorthy and Ernst (2001) and is called the *Feasibility Problem* where the aim is to merely find a feasible solution. This requires that each task is allocated to a qualified and available worker. It is also required that a worker cannot be assigned more than one task simultaneously as well as tasks cannot be pre-empted, meaning that each task has to be completed by one and the same worker.

In Table 2.1 attributes of PTSP variants are shown. The nomenclature of the attributes T, S, Q and O refer to the *Task type*, *Shift type*, *Qualifications* and *Objective function* respectively.

Table 2.1: PTSP variants

Attribute	**Type**	**Explanation**
T	F	Fixed contiguous tasks
	V	Variable task durations
	S	Split (non-contiguous) tasks
	C	Changeover times between consecutive tasks
S	F	Fixed, given shift lengths
	I	Identical shifts which are effectively of infinite duration
	D	Maximum duration without given start or end times
	U	Unlimited number of shifts of each type available
Q	I	Identical qualification for all staff (homogeneous workforce)
	H	Heterogeneous workforce
O	F	No objective, just find a feasible schedule
	A	Minimise assignment cost
	T	Worktime costs including overtime
	W	Minimise number of workers
	U	Minimise unallocated tasks

Many of the most basic and a few more complex problems can be described with this definition of PTSP attributes. It is, however, not possible to describe all of the numerous types of PTSP using these nomenclatures according to Krishnamoorthy and Ernst (2001).

By combining attributes it is possible to obtain more complex variants of the PTSP. An example would be the PTSP[F;F;H;A-T-W] mentioned in Krishnamoorthy and Ernst (2001), where multiple objectives are used. This problem has fixed contiguous tasks, fixed shift lengths, heterogeneous workforce and three objective functions (A-T-W), which represent assigment costs, work time with overtime included and requirements to minimize the number of workers respectively. The objective function for this problem is then a linear combination with parameters used to weigh (prioritize) them against each other.

Given the nomenclature above, our problem would be most related to the PTSP[F;F;H;F]. The difference is that the objective function is not empty. We are looking to maximize the number of qualified stand-ins each day as well as maximize employee satisfaction by meeting their recommendations. We have

a fixed number of workers, no costs and no unallocated tasks when a feasible solution is found. Therefore, none of the objective function types given in Table 2.1 are relevant in our case.

Some variants of the PTSP are given names in the literature, which is stated by Krishnamoorthy and Ernst (2001). An example is when the shifts and qualifications are identical (S=I, Q=I) and the objective function is to minimize the number of workers that are used (O=W). This variant, PTSP[F;I;I;W], has been published as the Fixed Job Schedule Problem and is described in Section 2.4.

2.1.1 Applications

An example where the PTSP can be found is when developing a rostering solution for ground personnel at an airport. This is mentioned in the article by Krishnamoorthy and Ernst (2001). The problem can be dealt with by first assigning the workers to days in order to satisfy all the labour constraints, followed by assigning the tasks to the scheduled workers.

In the article mentioned above, three further problems of type PTSP, related to airplanes are mentioned. They occur when scheduling for either airport maintenance staff, planes to gates or staff that do not stay in one location, such as airline stewards. Scheduling for airport maintenance staff can lead to either PTSP[F;I;H;U-A] or PTSP[F;I-U;H;W], which are similar problems but given two different names: Operational Fixed Interval Scheduling Problem and Tactical Fixed Interval Scheduling Problem respectively. These are both described further in Section 2.4.

Another application, which has been frequently studied, is classroom assignments and is discussed in Krishnamoorthy and Ernst (2001). Based on specifications such as the amount of students in a class or the duration of a class, different classrooms have to be considered. Requirements of equipment, e.g. for a laboratory, may also greatly limit the available classrooms to choose from. A majority of the complications of this problem is due to the fact that lessons can span over multiple periods.

Worth noting for classroom assignment problems is that there are no start or end times for the shifts, as they represent the rooms. The aim in the present problem would be to simply find a feasible assignment of classrooms. Therefore, the nomenclature of the problem would be PTSP[S;I;H;F], with the possibility of adding preferences to the objective function. An example of a preference would be to assign the lessons as close to each other as possible on a day, preventing travel distances for teachers and students.

2.2 Shift Minimisation Personnel Task Scheduling Problem

A close relative to the PTSP is the Shift Minimisation Personnel Task Scheduling Problem (SMPTSP), which is a variant where the aim is to minimize the cost occuring due to the number of personnel that are used. The same common traits for this problem are mentioned in the article Krishnamoorthy et al. (2012) as in the PTSP; workers with fixed work hours are to be assigned tasks with specified start and end times, for which they are qualified.

In the same article as above the authors "... concentrate mainly on a variant of the PTSP in which the number of personnel (shifts) required is to be minimised.". In doing so, it is possible to determine the lowest number of staff and the mix of staff a company need in order to be able to complete the tasks and still be operational. They also presumed that the pool of workers is unlimited for each skill group, which is not the case in the present problem due to the limitations of librarians and assistants.

Furthermore, it is said in Krishnamoorthy et al. (2012) that SMPTSP can be applied when there is a large number of workers available with different qualifications and it is needed to ensure that the tasks for that day are performed. The PTSP and SMPTSP are therefore useful day-to-day operational management tools and commonly occur in practical instances where tasks are allocated on a daily basis.

It is shown in Kroon et al. (1995) that SMPTSP is a complex problem even if the preemption constraint were to be removed. However, it is stated in Krishnamoorthy et al. (2012) that if the qualifications of the workers were identical it would be an easily solvable problem. The difference in publication year between the articles indicates that this problem has become less challenging in recent years.

During the last decade, a couple heuristics have been implemented to deal with the SMPTSP. One method introduced by Krishnamoorthy et al. (2012) is a Lagrangean relaxation approach that combines two problem specific heuristics: Volume Algorithm (VA) and Wedelin's Algorithm (WA). These heuristics exploit the special structure of the SMPTSP by relaxing some of the harder constraints into the objective function, thus being a problem specific heuristic. What remains after the relaxation is a problem decomposed into several independent problems which can be solved independently. A solution to these decomposed problems is discussed further in Krishnamoorthy et al. (2012).

Another way to solve the SMPTSP is to use a very large-scale neighbourhood search algorithm, as was presented by Smet and Vanden Berghe (2012). The main purpose of their implemented hybrid local search is to repeatedly fix and optimize to find neighbouring solutions. Using this method on 137 benchmark instances introduced by Krishnamoorthy et al. (2012), Smet and Vanden Berghe managed to find 81 optimal solutions compared to Krishnamoorthy et al. who only managed to find 67. Though, both methods found feasible solutions for 135 out of the 137 problem instances. This comparison is presented in Smet et al. (2014).

Smet et al. (2014) introduced a third and most effective method to solve the benchmark instances related to the SMPTSP. By using a versatile two-phase matheuristic approach, solutions to all 137 benchmark instances could be found for the first time. The procedure used in their implementation is to first generate an initial solution by using a constructive heuristic, followed by improving the solution using an improvement heuristic. Which constructive and improvement heuristic they considered can be seen in their article.

2.3 Tour Scheduling Problem

The Tour Scheduling Problem (TSP) is described by Loucks and Jacobs (1991) as a combination of shift scheduling and days-off scheduling. Shift scheduling

refers to creating sets of contiguous hours during which a worker is assigned for work. The need for days off scheduling typically occurs when the time horizon for scheduling is weekly or more and when weekend staff is needed. Using the notation used described in Section 2.1, this would be be classified as a variant of the PTSP[F;D;I;W] or PTSP[F;D;H;W], depending on if the workforce is homogeneous or heterogeneous.

According to Loucks and Jacobs (1991), the vast majority of all tour scheduling problems up to 1991 involve a homogeneous workforce, that is, any worker can perform any assigned task. One such early study of the scheduling problem often mentioned in literature is provided by Thompson (1988). The problem studied in this PhD thesis concern only homogeneous workforces and the task assignment part is not considered.

In the article by Loucks and Jacobs (1991), the authors study a tour scheduling problem with a heterogeneous workforce. The problem both involves tour scheduling and task assignment, where the latter part is most interesting to us. The problem is studied in the context of fast food restaurants, where certain personnel is qualified only for certain stations in the restaurant. In such industries, the demand of staff differs between weekdays and times of the day. Two worker attributes are considered, their availability for work and their qualifications to perform different tasks. The problem concerns finding shifts for all workers which are to be assigned a length between a minimum and maximum number of hours per day.

The representative problem studied in the article involves creating a one-week schedule for 40 workers in a fast food restaurant, available for eight different tasks with a seven-day and 128-hour workweek. Several synthetic problems are studied in the article; all, however, with a minimum shift length of three hours, a maximum shift length of eight hours and a maximum of five work days.

A similar problem to the one descibed by Loucks and Jacobs is studied by Choi et al. (2009). They focus on a particular fast food restaurant in Seoul, which is made a representative of fast food chains in general. In this study, only two types of workers are available, fulltime and part time workers, with no other reference to difference in skill. The different shifts are already given by the restaurant managers and the task is to combine them into a tour. The task assignment aspect is lacking in this article.

In both articles the main objective is to minimize both overstaffing and understaffing, which will both have economical consequences for the fast food chain. This is done by reducing or increasing the workforce. For a problem with a fixed workforce, such as ours, this objective function is not relevant. In the example studied by Loucks and Jacobs there is also a goal to meet staff demand on total working hours. This is modeled as a secondary goal and is similar to our goal and somehow models a "soft" value, which is of interest to us.

A more recent TSP concern monthly tour scheduling, as opposed to most literature which concerns only weekly scheduling. Such a study was done by Rong (2010). The main advantage of monthly scheduling over shorter time periods, as stated in the article, is the possibility to plan a schedule with respect to fairness and balance over a longer period of time. The problem concerns workers with different skills, where each worker also can possess multiple skills. This is referred to as a mixed skill problem. Thus, the problem is similar to our problem, where mixed skill is also present. In the study, workers have individual weekend-off requirements. The problem does not involve task assignment, which

makes it less relevant for us.

The solution methods used to solve the TSP differs greatly between the articles studied. In the older articles, such as Thompson (1988) and Loucks and Jacobs (1991), custom made algorithms very similar to the methods used in manual scheduling are proposed to solve the problem. These solution methods involve classifying staff and distributing them according to some rule (for example, the staff with the most scarce skill is assigned first). General commercial solvers are not proposed, due to lack of efficiency during these times.

In Hojati and Patil (2011), the same model and data is used as in Loucks and Jacobs (1991). Also this article states that commercial solvers are insufficient for solving the problem. Instead two methods are proposed, one which decomposes the problem into two problems solvable by commercial solvers and one customized heuristic method based on the Lagrangian relaxation method. This method solves the problem with more satisfactory results than in Loucks and Jacobs (1991) as explained in the article. In Choi et al. (2009), a pure integer programming method is used.

2.4 Other similar problems

In this section a couple of problems, similar to our own, will be described. The focus will not be on the variety of methods used to solve these problem; instead, the focus will be to give clarity to how closely related many of these problems are.

2.4.1 Fixed Job Schedule Problem

Variations of the task assignment problem relevant to our problem include for example the Fixed Job Schedule Problem (FJSP). The FJSP has been studied since the 1970s in the context of task assignment in processors. According to Krishnamoorthy et al. (2012), the problem concerns the distribution of tasks with fixed starting and ending times over a workforce with identical skills, such as processing units. Such problems have been solved by Gertsbakh and Stern (1978) and Fischetti et al. (1992).

In the article by Gertsbakh and Stern (1978) a situation is studied where n jobs need to be scheduled over an unlimited number of processors. The objective function of such a problem becomes to minimize the number of machines needed to perform all tasks. Fischetti solves a similar problem, but adds time constraints, saying that no processor is allowed to work for more than a fixed time T during a day as well as a spread time constraint forcing tasks to spread out with time gap S over a processor.

2.4.2 Tactical Fixed Interval Scheduling Problem

The Tactical Fixed Interval Scheduling Problem (TFISP) is a problem very closely related to the SMPTSP with the sole difference that the TFISP concerns workers which always are available, such as industrial machines or processors. The problem is studied in Kroon et al. (1995). A typical TFISP can be expressed using the nomenclature in Table 2.1 and written as PTSP[F;I-U;H;W].

Opposed to the FJSP, the TFISP deals with a heterogeneous workforce. Two different contexts are studied by Kroon et al. (1995). One of them concerns the handling of arriving aircraft passengers at an airport. Two modes of transport from the airplane to the airport are investigated, directly by gate or by bus. The two transportation modes thus correspond to two processing units, which can only handle a number of jobs at the same time.

2.4.3 Operational Fixed Interval Scheduling Problem

The Operational Fixed Interval Scheduling Problem (OFISP) is a close relative to TFISP, where both types are restricted by the following: Each machine (worker) cannot handle more than one job at a time, can only handle a subset of the jobs and preemption of jobs is not allowed. The difference between them occurs in the objective function, as described in Kroon et al. (1995). TFISP tries to minimize the number of workers, while OFISP tries to minimize the operational costs and the number of unallocated tasks using priority indices. In the present nomenclature this would give rise to the problem PTSP[F;I;H;U-A].

Given the problem definition above, working shifts are to be created for the workers and tasks have to be allocated on a day-to-day basis. OFISP, studied by Kroon et al. (1995), can therefore be seen both as a job scheduling problem and a task assignment problem.

2.4.4 Stochastic job problems

What differs mostly between the problem types described above and the problem studied in this thesis work, is the difference in objective function. The main objective function is often to minimize staff for a fixed number of jobs, not taking stand-in assignment into account.

An area where the need for stand-in personnel appears is in the maintenance industry, where some jobs can be foreseen and other (emergency) jobs are of a stochastic nature, that is, there is a probability that such jobs will occur a certain hour. The problem combing both foreseen and stochastic maintenance worker scheduling was studied by Duffuaa and Al-Sultan (1999), as a continuation of the work by Roberts and Escudero (1983).

In the article, a fixed, heterogeneous workforce consisting of electricians, plumbers and mechanics is studied. The shifts of the personnel are predetermined by their given work times and thus the problem becomes a pure task assignment problem. An objective function is used where the goal is to maximize the number of planned and unplanned jobs performed by the workers, by taking into account the probability of unplanned work to occur. Thus, certain workers will be left at the station as stand-ins in the case an unplanned job arises. The commercial solver LINDO was used to solve the problem.

2.5 Modeling soft constraints

For most scheduling problems, the main objective is to minimize worker-related costs by reducing the number of workers needed to perform a task, or by reducing the working time for part-time employees. Recently, however, many studies have started to focus more on softer values such as worker satisfaction as an objective

function. Such values are usually considered when scheduling is done manually, but have been forgotten or set aside in mathematical modeling.

In an article by Akbari et al. (2013) a scheduling problem for part-time workers with different preferences, seniority level and productivity is investigated. In this article, these aspects are reflected in the objective function and weighted against each other. A similar problem was also studied by Mohan (2008), but for a workforce of only part-time workers.

Other factors which may affect worker satisfaction, and in the long run efficiency and presence at work are fatigue, fairness and boredom. These are discussed by Eiselt and Marianov (2008). Repetitiveness of a job as well as the level of challenge can cause boredom of workers. Increasing variance in the schedule is done by Eiselt and Marianov (2008) through providing an upper bound of how many tasks can be performed in a given time span. The article suggests a sort of measurement of the distance between the task requirements and the worker abilities used. This will then be minimized in the objective function.

Another modeling method which is relevant specifically for scheduling problems featuring soft constraints is fuzzy goal programming. The method is discussed by Shahnazari-Shahrezaei et al. (2013), who model soft constraints as "fuzzy goals". These goals can become contradictory, for example could a preference of high seniority level workers come in conflict with a preference in working hours by an employee. The article uses fuzzy set theory, and uses a solution approach involving Li's two-phase method (Li (1990)). Soft constraints are modeled as trapezoid functions and an optimal solution with the best average value of all functions is found.

The solution methods proposed by Akbari et al. (2013) to solve a scheduling problem with soft constraints are two matheuristics: Simulated Annealing (SA) and Variable Neighbourhood Search (VNS). According to Akbari et al. (2013), SA has been studied as a solution method for the scheduling problem since the 1990s and many studies have shown that it is capable of providing near-optimal solutions in a short time compared to optimal integer-programming models for a variety of problems. An exponential cooling time was used for the algorithm and it was concluded that it was faster than the commercial solver LINGO in finding a solution.

VNS is the other proposed method by Akbari et al. (2013). A big difference between this and other methods is that VNS requires very little parameter tuning while often providing good solutions. Larger movement sizes are used at higher temperatures, and smaller in cooler temperatures. The method uses both a random and a systematic phase. In the random phase, worker schedules are regenerated randomly and better solutions are saved. In the systematic phase, two shifts are swapped. In the article, it was concluded that VNS could solve the problem faster than the implemented SA heuristic.

In the article by Eiselt and Marianov (2008), a commercial solver was used. This was also the case in Mohan (2008), although the article compares these results with the results obtained from a branch-and-cut algorithm. Also Shahnazari-Shahrezaei et al. (2013) uses a commercial solver.

2.6 Summary

In order to get an overview over the problems discussed in this chapter it is a good idea to look at general historical trends among the problems. One clear trend is the shift from problems concerning homogeneous to heterogeneous workforces. As stated by Krishnamoorthy et al. (2012), the problem of heterogeneous workforces was trivial at the publishing year, while in Loucks and Jacobs (1991), it is introduced as a rather new and challenging concept. Furthermore, the articles concerning the different PTSPs are mainly from the 1980s and 1990s and were challenging in their structure during these times. Some of them, such as FJSP, are related to the scheduling in processors, which was a hot topic at the start of the computer era. All the articles about modeling of soft constraints are presented after the year 2000, probably as a result of better computational powers.

The various solution methods found in the studied articles include commercial solvers, heuristics and matheuristics. Heuristics which have been studied in the articles include SA, VNS and Lagrangian relaxation, with and without VA. Also some local search-based heuristics have been used. Commercial solvers are used increasingly in newer articles such as Hojati and Patil (2011) and Mohan (2008), probably due to the improvement in performance in such solvers. Similarly, matheuristics are also discussed mostly in more recent articles, such as Akbari et al. (2013).

2.7 Relevance to our problem

As described in Section 2.1, many different types of personnel tasks scheduling problems exist. Only a few of them have been discussed in this chapter, of which some are closer related to our problem than others. In order to get a better understanding for how they are related to our problem, a few connections between between them will be presentedi n this section.

Using Table 2.1, the closest classification of the librarian scheduling problem studied in this thesis work would be the PTPS[F;F;H;F]. This describes a personnel task scheduling problem with fixed tasks, fixed shift lengths. a heterogeneous work force and an empty objective function. As stated in Section 2.1, the main difference lies in the objective function, as ours is not empty. Thus, many problem types discussed in this chapter concern problems which are similar to ours but with a different objective function. This includes the SMPTSP, the TSP, the FJSP, the TFISP and the OFISP. The first two are relevant to our problem as they concern task assignment and, in the case of TSP, days off scheduling. However, both involve shift scheduling while we have fixed shifts. The other three problems concern only task assignment, but are otherwise further from our problem since the tasks types are different from ours.

As stated in Section 2.1, our problem has an objective function in which the number of stand-in personnel is to be maximized. Such problems often arise in the maintenance job scheduling problem, as is discussed in Section 2.4.4. As a secondary objective, we are also interested in making the schedule varied and with a repeating pattern at an interval of five weeks. Modeling of such constraints is discussed in Section 2.5. The two sections are complementary to the previous sections, as they are less relevant in the overall problem formulation

but more relevant concerning the objective function.

Chapter 3

The mathematical model

In this chapter the mathematical model implemented to solve this problem will be presented. Prior to the objective function and constraints, the most significant sets and variables will be provided to give the reader some basic knowledge of the implemented model. Section 3.2 presents the objective function and gives a short description of what it represents. In Section 3.3 the essential constraints will be presented and explained. A complete model with all definitions and the full set of constraints can be found in Appendix A.

3.1 Set and variable definitions

To solve the problem many sets and variables had to be declared. There are many unique and individual requirements, given by the library, that have to be met. An example is that some personnel want a day free from outer tasks to be able to attend meetings or perform inner tasks. Another example is that some have two alternating schedules for odd and even weeks. These specific cases have to be modeled and result in a variety of set and variable definitions. Hence, only the most important ones are listed below. A complete list of the definitions can be found in Appendix A.

I	Set of workers
I_{lib}	Set of librarians ($I_{lib} \subseteq I$)
I_{ass}	Set of assistants ($I_{ass} \subseteq I$)
W	Set of all ten weeks
W_5	Set of first five weeks
D	Set of all days in a week
D_5	Set of all weekdays
S_d	Set of shifts available day d
S_3	Set of first three shifts on a weekday
J_d	Set of task types available day d

In order to further define the problem we introduce the following variables:

Let,

$$x_{iwdsj} = \begin{cases} 1, & \text{if worker } i \text{ is assigned in week } w, \text{ day } d, \text{ shift } s \text{ to a task } j \\ 0, & \text{otherwise} \end{cases} \tag{3.1}$$

$$H_{iwh} = \begin{cases} 1, & \text{if worker } i \text{ works weekend } h \ (= 1, 2) \text{ in week } w \\ 0, & \text{otherwise} \end{cases} \tag{3.2}$$

$$r_{iw} = \begin{cases} 1, & \text{if worker } i \text{ has its schedule rotated } w\text{-}1 \text{ steps} \\ 0, & \text{otherwise} \end{cases} \tag{3.3}$$

$$l_{iwd} = \begin{cases} 1, & \text{if librarian } i \text{ is a stand-in week } w, \text{ day } d \\ 0, & \text{otherwise} \end{cases} \tag{3.4}$$

$$a_{iwd} = \begin{cases} 1, & \text{if assistant } i \text{ is a stand-in week } w, \text{ day } d \\ 0, & \text{otherwise} \end{cases} \tag{3.5}$$

$$y_{iwds} = \begin{cases} 1, & \text{if worker } i \text{ works week } w, \text{ day } d, \text{ shift } s \text{ at task type E, I or P} \\ 0, & \text{otherwise} \end{cases} \tag{3.6}$$

$$W_{iwd} = \begin{cases} 1, & \text{if a worker } i \text{ is working a shift week } w, \text{ day } d \\ 0, & \text{otherwise} \end{cases} \tag{3.7}$$

$$b_{iw} = \begin{cases} 1, & \text{if worker } i \text{ works at H week } w \\ 0, & \text{otherwise} \end{cases} \tag{3.8}$$

$$f_{iw} = \begin{cases} 1, & \text{if worker } i \text{ is assigned to work friday evening week } w \\ 0, & \text{otherwise} \end{cases} \tag{3.9}$$

$$M_{wds} = \begin{cases} 1, & \text{if a big meeting is placed on week } w, \text{ day } d, \text{ shift } s \\ 0, & \text{otherwise} \end{cases} \tag{3.10}$$

$$m_{wdsD} = \begin{cases} 1, & \text{if a meeting is placed on week } w, \text{ day } d, \text{ shift } s \text{ at department } D \\ 0, & \text{otherwise} \end{cases} \tag{3.11}$$

$$d_{iwds} = \begin{cases} 1, & \text{if there is a difference in assignment of tasks at a shift} \\ & \quad s, \text{ for a worker } i, \text{ day } d \text{ between week } w \text{ and } w\text{+}5 \\ 0, & \text{otherwise} \end{cases} \tag{3.12}$$

$$l^{min} = \text{lowest number of stand-in librarians found (integer)} \tag{3.13}$$

$$a^{min} = \text{lowest number of stand-in assistants found (integer)} \tag{3.14}$$

Based on the variables defined above it has been possible to solve our scheduling problem. l^{min} and a^{min} are the variables of most significance as they represent the number of stand-ins found after a run.

3.2 Objective function

Due to multiple objective functions it has been necessary to weigh them against each other using parameters. These are shown in Equation 3.15 below as L, A and N.

$$maximize \quad L \cdot l^{min} + A \cdot a^{min} - N \cdot \sum_{i \in I} \sum_{w \in W_5} \sum_{d \in D_5} \sum_{s \in S_3} d_{iwds} \tag{3.15}$$

The two first objective functions represent the lowest amount of stand-in librarians and assistants found. The third objective function is a preference from the library that two weeks with a five-week interval should be as similar as possible (e.g. week 1 and 6, 2 and 7 etc.).

The parameter N prioritizes the similarity of weeks compared to the number of stand-ins. Based on the information given by the library it is a much higher priority to have many stand-ins. Hence, $N \ll L, A$ in our case.

If $L < A$ in the model the solver would prioritize assistants over librarians as stand-ins. Librarians are, however, more desired as stand-ins, due to their ability to perform all task types. Therefore, it is desired to set $L \geq A$.

3.3 Constraints

To model the present problem it has been of relevance to divide many of the constraints into weekend- and weekday constraints. Several help constraints have also been added to avoid multiplication of two variables, making the problem non-linear.

3.3.1 Demand and assignment constraints

The most crucial constraint is to ensure that the demand of workers is met each day. This can be modeled as:

$$\sum_{i \in I} x_{iwdsj} = demand_{wdsj}, \ \forall w \in W, d \in D, s \in S, j \in J_d \tag{3.16}$$

$demand_{wdsj}$ is an integer representing the number of workers required week w, day d, shift s for a task j.

The following constraint says whether a worker is assigned a task during a weekday where the evening shifts and task type *Library on Wheels* are excluded:

$$y_{iwds} = \sum_{j \in J_d \setminus \{B\}} x_{iwdsj}, \ \forall i \in I, w \in W, d \in D_5, s \in S_3 \tag{3.17}$$

The variable assignment above is used to simplify a couple of constraints.

To ensure that no worker is assigned more than one task the following constraint is implemented:

$$\sum_{s \in S} \sum_{j \in J_d} x_{iwdsj} \leq 1, \ \forall i \in I, w \in W, d \in D \tag{3.18}$$

However, if we allow a person to have two shifts at the Library on Wheels on a day, Equation 3.18 has to be slightly modified, which is shown in the complete model in Appendix A.

It is preferred to allow only one P per week and a maximum of three P per ten weeks. These are easily modeled with the following constraints:

$$\sum_{s \in S_d} \sum_{d \in D} x_{iwdsP} \leq 1, \ \forall i \in I, w \in W \tag{3.19}$$

$$\sum_{w \in W} \sum_{s \in S_d} \sum_{d \in D} x_{iwdsP} \leq 3, \ \forall i \in I \tag{3.20}$$

The duration of a P, which is for an entire day, is the cause of these preferences. Some workers are required to have some alloted time to perform inner tasks.

Another preference is to have various start times of the assigned tasks each week, so that the more and less desired shifts are evenly distributed. The following equation was implemented to meet such requirements:

$$\sum_{d \in D_5} y_{iwds} \leq 2, \ \forall i \in I, w \in W, s \in S_3 \tag{3.21}$$

Equation 3.21 allows a worker to have at most two tasks starting at the same hour. Worth noting is that task type *Library on Wheels* is disregarded in the constraint as the variable y_{iwds} is used; see Equation 3.17 for the definition.

It is desirable to avoid assigning too many tasks to a worker. Reason for this is to let the worker have some time to allot for inner services during weekdays. The equation below models this preference:

$$\sum_{d \in D_5} \sum_{s \in S_d} \sum_{j \in J_d} x_{iwdsj} \leq 4, \ \forall i \in I, w \in W \tag{3.22}$$

Equation 3.22 allows a worker at most four shifts per week.

3.3.2 Weekend and rotation constraints

To further model the problem weekends and week rotations have to be considered. The following three constraints are the most basic constraints regarding weekends H_{iwh} and week rotations r_{iw}:

$$\sum_{w \in W} r_{iw} = 1, \ \forall i \in I \tag{3.23}$$

$$\sum_{w \in W} H_{iwh} \leq 1, \ \forall i \in I, h = 1, 2 \tag{3.24}$$

$$r_{iw} \geq H_{iw1}, \ \forall i \in I, w \in W \tag{3.25}$$

Equation 3.23 provides all workers with a rotation of their schedule regardless if they are working weekends or not. Equation 3.24 allows a worker a maximum of two weekends ($h = 1, h = 2$) per ten weeks. Lastly, Equation 3.25 in combination with Equation 3.23 ensures that the schedule rotation is aligned with the first weekend, if the worker is due for weekend work. In case the worker is not due for weekend work, the rotation of the schedule is free.

A worker is supposed to work weekends with a five-week interval. However, in case when there are enough workers to satisfy the demand on weekends it may be enough to work one weekend per ten weeks for some. To avoid problems with the rotation when only the second weekend is assigned to a worker, the following equation was implemented:

$$r_{i(10(w+4)+1)} \geq H_{iw2},\ \forall i \in I, w \in W \tag{3.26}$$

Worth noting is that if a worker is assigned both weekends then Equation 3.26 provides the same information as Equation 3.25.

Every workers schedule is able to rotate up to nine times, where the decision variable r_{iv} decides the rotation. Therefore, the parameter $qualavail_{iwdsj}$ has to align with the rotation so that all workers are assigned tasks only when available. This is provided in the following equation:

$$x_{iwdsj} \leq \sum_{v \in V} r_{iv} * qualavail_{i(mod_{10}(w-v+10)+1)dsj},\ \forall i \in I, w \in W, d \in D, s \in S_d, j \in J_d \tag{3.27}$$

Table 3.1 gives a better understanding of how the modulus function in the parameter $qualavail_{iwdsj}$ is used. w represent the current week in the relative schedule and v represent *v-1* rotations to the right from the relative schedule. The resulting week after rotations can be seen in the cells.

Table 3.1: Resulting table of the function $mod_{10}(w - v + 10) + 1$

		w =									
		1	2	3	4	5	6	7	8	9	10
v =	1	1	2	3	4	5	6	7	8	9	10
	2	10	1	2	3	4	5	6	7	8	9
	3	9	10	1	2	3	4	5	6	7	8
	4	8	9	10	1	2	3	4	5	6	7
	5	7	8	9	10	1	2	3	4	5	6
	6	6	7	8	9	10	1	2	3	4	5
	7	5	6	7	8	9	10	1	2	3	4
	8	4	5	6	7	8	9	10	1	2	3
	9	3	4	5	6	7	8	9	10	1	2
	10	2	3	4	5	6	7	8	9	10	1

An example to better understand the function: Imagine that we are looking at an unrotated relative schedule where everyone is said to work weekend the first week. Say that we look at a workers first week, $w = 1$. If this schedule is rotated two times to the right, $v = 3$, then the first week in the new rotated schedule represent the previous ninth week, $w = 9$ from the old schedule.

A worker is supposed to work with the same task both Saturday and Sunday when working a weekend. This can be modeled with the following constraints:

$$\sum_{j \in J_d} x_{iw61j} + \sum_{j \in J_d} x_{iw71j} = 2 * \sum_{h=1}^{2} H_{iwh},\ \forall i \in I, w \in W \tag{3.28}$$

$$x_{iw61j} = x_{iw71j},\ \forall i \in I, w \in W, j \in J_d \tag{3.29}$$

Friday evening is also included in extension to working Saturday and Sunday, unless the worker is assigned to HB. It is, however, not a necessity to perform the same task Friday evenings as during the weekend, thus Fridays not included in Equation 3.29. Equation 3.30 below adds Fridays to the weekend:

$$\sum_{j \in J \setminus \{B\}} x_{iw54j} = f_{iw}, \ \forall i \in I, w \in W \tag{3.30}$$

The variable f_{iw} is, as mentioned in the variable declaration, equal to one if and only if a worker is working weekend as well as not being assigned to HB. The help constraints implemented to avoid multiplication of the two variables b_{iw} and H_{iwh} are left out of this section and can instead be seen in Appendix A.

It is of interest to implement a constraint preventing workers being assigned to HB more than once every ten weeks to avoid unfairness. The constraint can be modeled as:

$$\sum_{w \in W} \sum_{d=6}^{7} x_{iwd1HB} \leq 2, \ \forall i \in I \tag{3.31}$$

The combination of Equations 3.31 and 3.29 ensure that once a worker is assigned to HB it will be for two consecutive weekend days, which is the amount of days the previous equation allow a worker every ten weeks. Why this is preferable is described in Section 1.2.1.

3.3.3 Objective function constraints

To calculate the variable l^{min} and a^{min} used in the objective function (3.15) the following equations were implemented:

$$l^{min} \leq \sum_{i \in I_{lib}} l_{iwd}, \ \forall w \in W, d \in D_5 \tag{3.32}$$

$$a^{min} \leq \sum_{i \in I_{ass}} a_{iwd}, \ \forall w \in W, d \in D_5 \tag{3.33}$$

l_{iwd} and a_{iwd} are binary variables stating, for a day d, whether a worker is a stand-in or not. l^{min} and a^{min} are being maximized in the objective function; therefore, they will assume the lowest value of stand-in librarians and assistants respectively, for any day during the ten weeks. Just as with previous equations, help constraints have been left out for simplicity reasons. In this case regarding how l_{iwd} and a_{iwd} are determined.

As stated in Section 3.2, two weeks with a five-week interval should be as similar as possible. Equation 3.34 in combination with the Objective function 3.15 provides this preference:

$$d_{iwds} = |y_{iwds} - y_{i(w+5)ds}|, \ \forall i \in I, w \in W_5, d \in D_5, s \in S_3 \tag{3.34}$$

The decision variable d_{iwds} in the previous equation states if there is a difference in assignment between two tasks at the same hour and day for two weeks: w and $w + 5$. Thus, a variable minimized in the objective function.

3.3.4 Meeting constraints

At the library there are both library meetings as well as department meetings, where both occur with a five-week interval. Library meetings are set to take place from 8-10 on Mondays, whereas department meetings are more freely distributed. A few workers are not assigned library meetings, since they are needed at the stations to keep the library running. The set I_{big} in the equation below consists of all workers who are to be assigned library meetings. The constraints modeling library meetings are as follows:

$$\sum_{w \in W_5} M_{w11} = 1 \tag{3.35}$$

$$M_{(w+5)11} = M_{w11},\ \forall w \in W_5 \tag{3.36}$$

$$\sum_{s=2}^{3} \sum_{j \in J_d \setminus \{B\}} x_{iw1sj} \leq 1 - M_{w11},\ \forall i \in I \setminus I_{big}, w \in W \tag{3.37}$$

Equation 3.35 and 3.36 assign two library meetings with a five-week interval during the ten-week scheduling period. Equation 3.37 makes sure that the workers that do not attend library meetings are assigned any other task the day of the meeting.

The constraints added to model department meetings are somewhat similar to the library meeting constraints. Just as for library meetings they take place two times during the ten weeks with a five-week interval, which is described by Equation 3.38 and 3.39.

$$\sum_{w \in W_5} \sum_{d \in D_5} \sum_{s \in S_3} m_{wdsD} = 1 \tag{3.38}$$

$$m_{(w+5)dsD} = m_{wdsD},\ \forall D = 1, \ldots, 3, w \in W_5, d \in D_5, s \in S_3 \tag{3.39}$$

$$m_{wdsD} + x_{iwdsj} \leq 1,\ \forall D = 1, \ldots, 3, i \in I_D, w \in W, d \in D_5, s \in S_3, j \in J_d \tag{3.40}$$

$$m_{wdsD} \leq \sum_{v \in V} r_{iv} * qualavail_{i(mod_{10}(w-v+10)+1)dsE},\ \forall D = 1, \ldots, 3, i \in I_D, w \in W, d \in D_5, s \in S_3 \tag{3.41}$$

Equation 3.40 prohibits a worker from multitasking, i.e. attend a meeting as well as work with a task. Lastly, Equation 3.41 enables department meetings only when everyone in that department is available for a task; the rotation is taken into the account.

Chapter 4

Results from mathematical model

4.1 Problem studied

An adaptation of the problem described ni the problem description was modeled and run in CPLEX. The special case studied involved the specific demands of the library on the personnel such as who is not particularly good at a task. For this reason, many extra constraints were added, see Appendix A. The features of the problem studied involve:

- A schedule of 10 weeks was to be generated.

Running the mathematical model in CPLEX 12.5.0.0 finds an optimum solution in approximately 17 minutes (1031). In the objective function, the parameters L = 4000, A = 2000 and N = 5 were used, thus having twice as much priority on finding librarian stand-ins than assistant stand-ins.

4.2 Evaluation of results

Chapter 5

Task distribution approach

5.1 Introduction

The first approach, where fixed weeks are distributed to workers, is discussed in the previous chapter. The second approach, which is presented in this chapter, instead distributes individual tasks to workers. This method greatly resembles the process of manually placing tasks as is typically done in many practical situations.

The objective of the scheduling process is not only to schedule a number of tasks to the workers but also to place them optimally with respect to stand ins. Thus, a method for moving through the solution space and a way of distinguishing between good and bad solutions is of great importance.

The primary method used in this approach is a large neighbourhood search (LNS) together with a simulated annhealing (SA) accept function. Destroying and repairing the solution, as is customary in LNS, helps leading the solution out of local optima or plateaus. Similarly, SA is used in order to allow the solution to move in a less favourable directions to avoid these local optima. The search is guided by a continuously updated schedule cost.

Write: two phases implemented.

5.2 Objective functions

Distinguishing good schedules from bad schedules means there is a need for a way of seting a cost to different schedules. In the original mathematical model, the cost of a schedule consists of two terms: a weighted sum of the number of stand ins on the worst day and the number of different shifts present in the schedule. The rest of the model consists of hard constraints which cannot be violated. However, in the heuristic approach these hard constraints are divided into hard and soft constraints, as is illustrated in Figure (TODO).

During the scheduling process in the implementation, three different objective functions are used. The objective functions are illustrated in Table (TODO) and consist of a weekend objective function, a worker objective function and a weekday objective function.

The weekday objective function is associated with the weekend distribution phase of the problem. In the weekend objective function, a stand in cost is

Table 5.1: Objective functions used in the implementation.

Weekend Objective Function	Min stand in cost + average num stand ins Min shift availability cost + average shift avail Min day availability cost + average day avail
Worker Objective Function	Num tasks per day cost Num tasks per week cost Num PL per week cost Total num of PL cost Num tasks at same shift per week cost
Weekday Objective Function	Min stand in cost

defined as the weighted sum between the number of librarians and assistants which are stand ins at a certain day. The minimum stand in cost is defined as the cost at the worst av all days throughout all weeks.

The min shift availability cost refers to the minimum number of workers available at a shift throughout all shifts, days and weeks. Similarly, the min day availability cost is the lowest number of workers available any shift throughout the whole schedule. In addition to these costs, the average of all three cost types is also used to distinguish solutions which have the same cost.

The worker objective function is used during the weekday task distribution phase of the problem. The function value is a combination of the relaxed constraints of the problem, as described at the beginning of this chapter. Thus, the worker objective function must always be reduced to zero before the schedule will be considered feasible.

The objective function consists of five different costs. All of them are calculated as the total cost for all workers. For example, the first cost described in Table (todo) is the total number of tasks per day exceeding the maximum of one daily task. In the objective function, the sum of excess tasks is taken over all workers.

The second cost is the number of tasks exceeding four per week. The third is the number of Fetch list tasks performed by a worker exceeding the max limit of one per week. In addition to this, there is a specified max limit for the number of fetch lists that can be performed by a worker throughout all weeks and which is different for different workers. This is the fourth cost. Finally, the number of tasks performed at the same shift in a week should not exceed two, which is the final cost in the worker objective function.

A feasible solution, where the worker objective function is zero, is evaluated using the weekday objective function described lastly in the table. This cost has only one cost and corresponds to the objective function in the mathematical model. The min stand in cost is calculated in the same way as in the weekend objective function.

5.3 Weekend phase

The first phase in the scheduling process is the weekend phase. In this phase the weekends of all workers are placed and optimized before placing the remaining tasks. The reason for implementing such a phase rather than placing all tasks at once was the big impact of the weekend structure on the entire schedule. First and foremost, the location of a worker's weekend affects the availability of the worker in the following week where the week rest is placed. The worker is unavailable during week rest and if such week rests are combined in an unfortunate way, the scheduling can result in an uneven distribution of workers during days affected by week rest.

In order to measure what a good distribution of weekends is, the weekend objective function described in the previous section is used during the search process. Although the overall objective is to maximize the number of stand ins at the most critical day in the schedule, that is, even out the stand ins over the days, it is not trivial to measure this in a schedule with no tasks placed. Evening work reduces potential stand ins as well as tasks distributed during the days. Furthermore, limits on how many tasks per week a person is allowed to take further complicates the measurement of stand ins.

Because of the difficulty in measuring stand ins, certain tasks are placed already in the weekend phase. This includes all Library on Wheel tasks and all evening tasks. For the Library on Wheels, there is not much choice or variability, and thus a fixed schedule is used. Similarly, the workers who are available at the evening tasks are in most part, equal to the number of workers needed, thus leaving little choice in scheduling evenings. This is illustrated in tables 5.2 and 5.3.

Table 5.2: Worker availability placing only weekends.

	Num available assistants						
	Mo	Tu	We	Th	Fr	Sa	Su
Shift 1:	8	10	10	10	6	0	0
Shift 2:	8	9	9	9	6	0	0
Shift 3:	9	8	9	7	5	0	0
Shift 4:	3	2	2	3	0	0	0
	Num available librarians						
	Mo	Tu	We	Th	Fr	Sa	Su
Shift 1:	16	15	16	13	12	0	0
Shift 2:	18	15	17	14	13	0	0
Shift 3:	17	14	18	18	13	0	0
Shift 4:	3	4	4	3	1	0	0
	Num available BBlib						
	Mo	Tu	We	Th	Fr	Sa	Su
Shift 1:	2	0	1	1	1	0	0
Shift 2:	0	0	0	0	0	0	0
Shift 3:	0	0	0	0	0	0	0
Shift 4:	1	0	2	2	0	0	0

The the number of available workers present at a shift, as illustrated in the

Table 5.3: Worker availability after placing weekends as well as evening tasks and BokB for the same week.

	Num available assistants						
	Mo	Tu	We	Th	Fr	Sa	Su
Shift 1:	7	10	9	9	8	0	0
Shift 2:	7	9	8	8	8	0	0
Shift 3:	7	7	8	6	6	0	0
Shift 4:	0	0	0	0	0	0	0

	Num available librarians						
	Mo	Tu	We	Th	Fr	Sa	Su
Shift 1:	14	11	13	11	12	0	0
Shift 2:	14	11	14	11	12	0	0
Shift 3:	13	11	14	13	13	0	0
Shift 4:	0	0	0	1	1	0	0

	Num available BBlib						
	Mo	Tu	We	Th	Fr	Sa	Su
Shift 1:	0	0	0	0	0	0	0
Shift 2:	0	0	0	0	0	0	0
Shift 3:	0	0	0	0	0	0	0
Shift 4:	0	0	1	0	0	0	0

tables, includes all those workers who are available for tasks at that shift and who has no tasks scheduled at that day.

The solution process follows an algorithm which involves both LNS and SA.

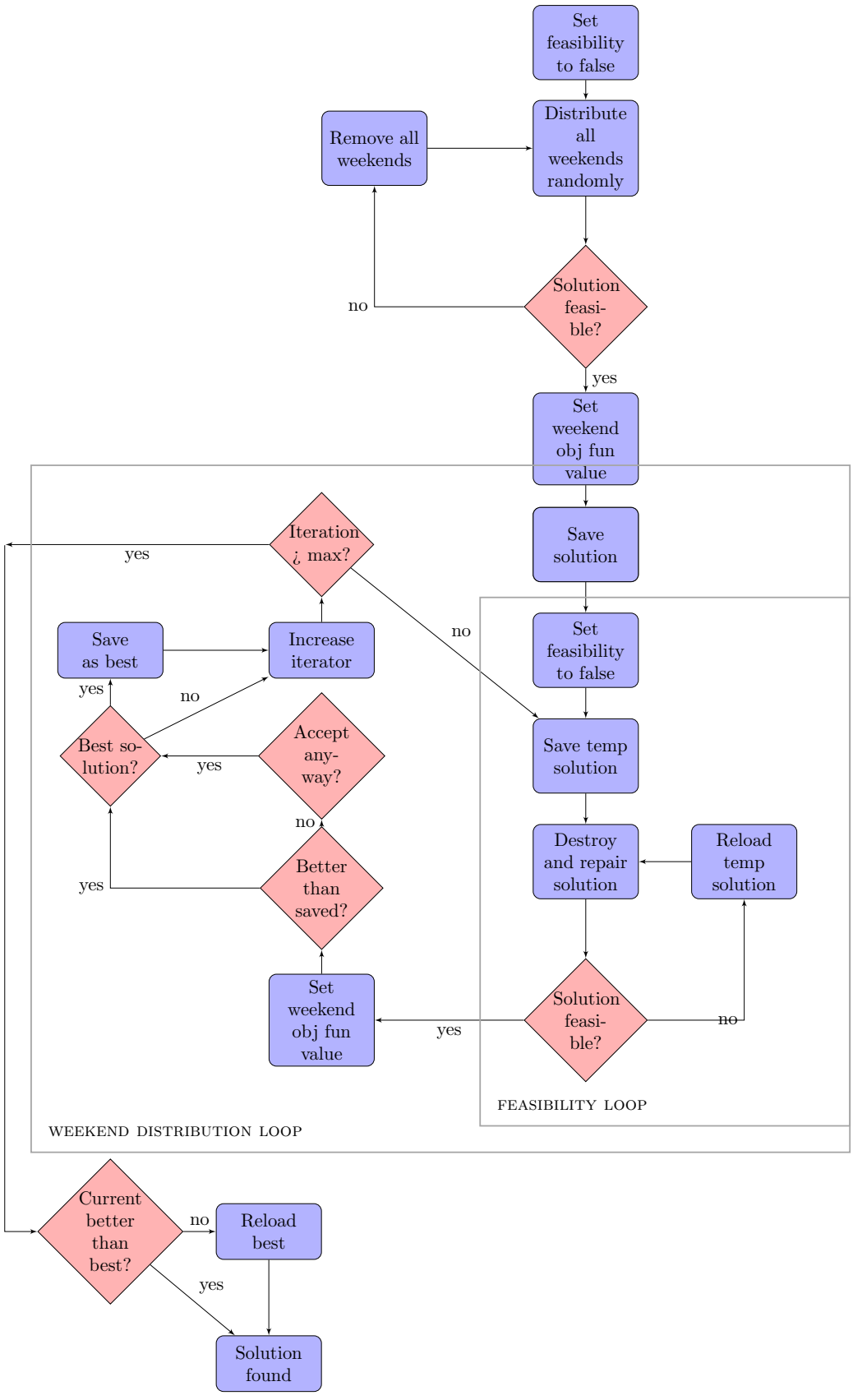
Table 5.4: Algorithm for solving the weekend schedule.

Step :	Set feasibility to false.	
Step :	**while**(infeasible)	
Step 1:		Distribute all weekends at random to workers.
Step 2:		Check feasibility of solution.
Step :	Set weekend objective function value	
Step :	**for**(N iterations)	
Step :	Save current solution	

Feasibility check:

Initial solution: placing tasks at random Destroy: Destroying a certain number of weekends at random. Repair: Repairing same weekends for same workers. Prioritizing weekends according to 1. qualification 2. avail demand diff. Almost random worker placed, although making sure that HB is placed correctly.

Infeasibility check: Does the current assignment of weekends generate a


Set feasibility to false
Distribute all weekends randomly
Remove all weekends
Solution feasi-ble?
no
yes
Set weekend obj fun value
Save solution
Iteration ¿ max?
yes
no
Set feasibility to false
Save as best
Increase iterator
yes
no
Save temp solution
Best so-lution?
Accept any-way?
yes
no
Destroy and repair solution
Reload temp solution
yes
Better than saved?
Set weekend obj fun value
Solution feasi-ble?
yes
no
FEASIBILITY LOOP
WEEKEND DISTRIBUTION LOOP
Current better than best?
no
Reload best
yes
Solution found

schedule where there are not enough workers at the shifts?

Evenings: based on worker costs.

Heuristic methods: SA on LNS with random destroy and repair. SA accept function, accepting with exponential cooling. Tuning parameters T and alpha.

5.4 Weekday phase

Evenings,weekends and BokB already placed.

Concept: destroy worst worker until all workers have feasible schedules. Record the library cost of the solution.

Destroy: weekday tasks for workers with highest cost. Repair: 1. qualification, 2. avail demand diff. Place cheapest worker.

Infeasibility: when a feasible worker cost is not found for a large number of iterations.

5.5 Simplifications of Mathematical Model

-10 Week scheduling -Objective function term about similar weeks -BokB fixed weeks for every other week workers. -Even odd weeks. How to handle? (-lower limit PL)

5.6 Implementation

C++, object orientation, run on a linux operating system. Reading availability of workers into the program and outputing a result file, which can be read by Excel. Results are to be visualized in Excel (write this in another part?)

Bibliography

A. T. Ernst, H. Jiang, M. Krishnamoorthy, B. Owens, and D. Sier. An annotated bibliography of personnel scheduling and rostering. *Annals of Operations Research*, 127(1-4):21–144, 2004.

M. Krishnamoorthy and A. T. Ernst. The personnel task scheduling problem. In *Optimization methods and applications*, pages 343–368. Springer, 2001.

M. Krishnamoorthy, A. T. Ernst, and D. Baatar. Algorithms for large scale shift minimisation personnel task scheduling problems. *European Journal of Operational Research*, 219(1):34–48, 2012.

L. G. Kroon, M. Salomon, and L. N. Van Wassenhove. Exact and approximation algorithms for the operational fixed interval scheduling problem. *European Journal of Operational Research*, 82(1):190–205, 1995.

P. Smet and G. Vanden Berghe. A matheuristic approach to the shift minimisation personnel task scheduling problem. *Practice and theory of automated timetabling*, pages 145–151, 2012.

P. Smet, T. Wauters, M. Mihaylov, and G. Vanden Berghe. The shift minimisation personnel task scheduling problem: A new hybrid approach and computational insights. *Omega*, 46:64–73, 2014.

J. S. Loucks and F. R. Jacobs. Tour scheduling and task assignment of a heterogeneous work force: A heuristic approach. *Decision Sciences*, 22(4):719–738, 1991.

G. M. Thompson. *A comparison of techniques for scheduling non-homogeneous employees in a service environment subject to non-cyclical demand volume I, chapters l-7*. PhD thesis, The Florida State University, 1988.

K. Choi, J. Hwang, and M. Park. Scheduling restaurant workers to minimize labor cost and meet service standards. *Cornell Hospitality Quarterly*, 50(2): 155–167, 2009.

A. Rong. Monthly tour scheduling models with mixed skills considering weekend off requirements. *Computers & Industrial Engineering*, 59(2):334–343, 2010.

M. Hojati and A. S. Patil. An integer linear programming-based heuristic for scheduling heterogeneous, part-time service employees. *European Journal of Operational Research*, 209(1):37–50, 2011.

I. Gertsbakh and H. I. Stern. Minimal resources for fixed and variable job schedules. *Operations Research*, 26(1):68–85, 1978.

M. Fischetti, S. Martello, and P. Toth. Approximation algorithms for fixed job schedule problems. *Operations Research*, 40(1-supplement-1):S96–S108, 1992.

S. O. Duffuaa and K. S. Al-Sultan. A stochastic programming model for scheduling maintenance personnel. *Applied Mathematical Modelling*, 23(5):385–397, 1999.

S. M. Roberts and L. F. Escudero. Scheduling of plant maintenance personnel. *Journal of Optimization theory and Applications*, 39(3):323–343, 1983.

M. Akbari, M. Zandieh, and B. Dorri. Scheduling part-time and mixed-skilled workers to maximize employee satisfaction. *The International Journal of Advanced Manufacturing Technology*, 64(5-8):1017–1027, 2013.

S. Mohan. Scheduling part-time personnel with availability restrictions and preferences to maximize employee satisfaction. *Mathematical and Computer Modelling*, 48(11):1806–1813, 2008.

H. A. Eiselt and V. Marianov. Employee positioning and workload allocation. *Computers & operations research*, 35(2):513–524, 2008.

P. Shahnazari-Shahrezaei, R. Tavakkoli-Moghaddam, and H. Kazemipoor. Solving a multi-objective multi-skilled manpower scheduling model by a fuzzy goal programming approach. *Applied Mathematical Modelling*, 37(7):5424–5443, 2013.

R. J. Li. *Multiple objective decision making in a fuzzy environment.* PhD thesis, Kansas State University, 1990.

Appendix A

Problem definitions

A.1 Sets

I	Set of workers
I_lib	Set of librarians (I_lib $\subseteq$ I)
I_ass	Set of assistants (I_ass $\subseteq$ I)
W	Set of weeks
D	Set of days in a week
S_d	Set of shifts day d
J_d	Set of task types day d
I_LOW	Set of librarians available to work in library on wheels
I_free_day	Set of workers that shall be assigned a free weekday per week
I_odd_even	Set of all workers with odd or even weeks
I_weekend_avail	Set of workers available for weekend work

A.2 Variables

$$x_{iwdsj} = \begin{cases} 1, & \text{if worker } i \text{ is assigned in week } w, \text{ day } d, \text{ shift } s \text{ to a task } j \\ 0, & \text{otherwise} \end{cases} \tag{A.1}$$

$$H_{iwh} = \begin{cases} 1, & \text{if worker } i \text{ works weekend h in week } w \\ 0, & \text{otherwise} \end{cases} \tag{A.2}$$

$$r_{iw} = \begin{cases} 1, & \text{if worker } i \text{ has its scheduled rotated } \textit{w-1 steps} \\ 0, & \text{otherwise} \end{cases} \tag{A.3}$$

$$lib_{iwd} = \begin{cases} 1, & \text{if librarian } i \text{ is a stand-in week } w \text{ day } d \\ 0, & \text{otherwise} \end{cases} \tag{A.4}$$

$$ass_{iwd} = \begin{cases} 1, & \text{if assistant } i \text{ is a stand-in week } w \text{ day } d \\ 0, & \text{otherwise} \end{cases} \tag{A.5}$$

$$y_{iwds} = \begin{cases} 1, & \text{if worker } i \text{ is working } w \text{ day } d \text{ regardless of task type} \\ 0, & \text{otherwise} \end{cases} \tag{A.6}$$

$$hb_{iw} = \begin{cases} 1, & \text{if assistant } i \text{ is a stand-in week } w \text{ day } d \\ 0, & \text{otherwise} \end{cases} \tag{A.7}$$

$$friday\_evening_{iw} = \begin{cases} 1, & \text{if assistant } i \text{ is a stand-in week } w \text{ day } d \\ 0, & \text{otherwise} \end{cases} \tag{A.8}$$

$$lib\_min = \text{lowest number of stand-in librarians found (integer)} \tag{A.9}$$

$$ass\_min = \text{lowest number of stand-in assistants found (integer)} \tag{A.10}$$

A.3 Parameters

$$N1l = \text{a value to prioritize the amount of stand-in librarians} \tag{A.11}$$

$$N1a = \text{a value to prioritize the amount of stand-in assistants} \tag{A.12}$$

$$N2 = \text{a value to prioritize similar weeks} \tag{A.13}$$

$$avail\_day_{iwd} = \begin{cases} 1, & \text{if worker } i \text{ is available for work week } w, \text{ day } d \\ 0, & \text{otherwise} \end{cases} \tag{A.14}$$

$$task\_demand_{dsj} = \text{number of workers required day } d, \text{ shift } s \text{ for task type } j \tag{A.15}$$

$$qualavail_{iwdsj} = \begin{cases} 1, & \text{if worker } i \text{ is qualified and available week } w, \text{ day } d, \text{ shift } s \text{ for task type } j \\ 0, & \text{otherwise} \end{cases} \tag{A.16}$$

$$LOW\_demand_{wds} = \text{number of workers required day } d, \text{ shift } s \text{ at the library on wheels} \tag{A.17}$$