

MASTER

Cyclical personnel scheduling using column generation

Meijer, M.C.C.

Award date:
2018

[Link to publication](#)

Disclaimer

This document contains a student thesis (bachelor's or master's), as authored by a student at Eindhoven University of Technology. Student theses are made available in the TU/e repository upon obtaining the required degree. The grade received is not published on the document as presented in the repository. The required complexity or quality of research of student theses may vary by program, and the required minimum study period may vary in duration.

General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain

Cyclical personnel scheduling using column generation

Author

Coen Meijer

St.nr. 0782798

Ist Supervisor TU/e
2nd Supervisor TU/e
Company supervisor
Company supervisor

dr.ir. N.P. Dellaert
dr. M. Firat
F. Dommers
ir. F. van Jeveren

Date

November 1, 2018

TUE. School of Industrial Engineering.
Series Master Theses Operations Management and Logistics

Subject headings: personnel scheduling, column generation, days off scheduling, network flow modelling, security guards scheduling.

Preface

This research is the final step for me to complete the Master in Operations Management and Logistics and marks the end of an important part of my life. Conducting this project has not always been without difficulties, but there were always a lot of people to support me. They helped me bring this project to a success which would not have been possible without them and I would like to take this opportunity to thank them.

First of all, I want to thank my girlfriend for always supporting me, for being my cheerleader and for pushing me when needed. Next, I am grateful to my parents for giving me the opportunity to complete a Master education and for always being there. Of course, I want to thank my friends and family, they always lent me a listening ear and I could always laugh with them. A special thanks goes to Joost, Sandra and Maudy. My brother Joost shared his experience with graduating and gave me good advice throughout this project. Sandra has helped me a lot to find a company to conduct my research at by bringing me into contact with her managers at EBN. Maudy has put a lot of effort into proofreading this thesis and helped me to bring my grammar and sentencing skills in English to a higher level.

I am very happy that Nico wanted to supervise my graduation project as first supervisor and that Nico has brought Murat to the team as a second supervisor. It was a great experience to work with Nico and Murat and to see their expertise. They always understood exactly the problems I encountered and how to guide me to find a solution or improvement. I want to thank Nico and Murat for their enthusiasm and effort during this project and for sharing their tremendous knowledge.

Last but definitely not least, I want to greatly thank my supervisors at EBN, Freddy and Frank, for giving me the opportunity to conduct my graduation project at EBN. They have provided me with a great environment to carry out my research, by giving me a lot of freedom and providing me with all the information and data I needed. Moreover, I have been welcomed very friendly at EBN and have been helped by a lot of colleagues. I want to thank all colleagues at EBN for answering my questions and for making me feel part of the team.

I have worked on this project with great enthusiasm and found it very enjoyable to build the model step by step and to see the results improve. Furthermore, it was very educational to work at a planning department where I could experience and see a lot of operational planning issues. Working towards my graduation has been fun and interesting and I am excited to put my knowledge into practice and to start a job in a business environment.

Abstract

This research developed an algorithm to construct cyclical schedules for security guards. Important characteristics of the model are that work is 24/7 and that skill levels are discrete. This means that days off must be scheduled and that every duty requires a unique skill. It was first attempted to model the situation using a network flow model where duties and rest periods are represented by arcs. However, this was too complex to solve within reasonable time on a desktop PC. Therefore, a column generation scheme was applied to find a good solution to the scheduling problem. The column generation scheme consists of a sub problem and a master problem. The sub problem constructs schedules for one employee at a time using the network flow model. The master problem selects a schedule for every employee from the schedule pool generated by the sub problem.

Management summary

In this chapter the goals and results of this research are summarized without going into technical details. First, a brief introduction of the research project is given. Thereafter, the research question and a selection of the sub-questions will be stated and answered.

Introduction

This research project was conducted at EBN and is the final step in completing my Master education. EBN is a company located in Breda that offers security services to its clients. Every month, EBN carries out more than 2000 duties and employs around 150 employees to achieve this. In order to make sure that employees are at the right place at the right time and that every duty is carried out by a suitable employee, schedules are constructed. As one can imagine it is a complex task to make a schedule containing that many elements. Currently, the planning department constructs these schedules manually. Management of EBN was looking for a way to help the planning department and wanted to improve the quality of their schedules. Therefore, this research project was initiated to develop an algorithm that generates schedules of better quality in a less labor-intensive fashion.

Research question

How can the scheduling process at EBN be improved by developing and implementing a scheduling tool that applies an optimization algorithm?

Finding a feasible solution to the scheduling problem is a difficult task. There are many rules and agreements coming from legislation, labor agreements and preferences that have to be taken into account. Especially the fact that an adjustment in one employee's schedule can have extensive implications for schedules of other employees, makes the scheduling problem a complex task. For the planning department it is time consuming to construct a feasible aggregate schedule that covers all duties and is in accordance with legislation, let alone to optimize the schedule for preference rules and costs. To help the planning department, an algorithm was developed that can be executed by a computer program called AIMMS. The algorithm is able to generate feasible schedules and is able to optimize for costs including penalties for disregarding preference rules. Result of this research project is a scheduling tool that can generate schedules without violating legislation and preference rules and is able to minimize overtime and undertime. This is an improvement to the current situation, because, as described in chapter 7, total costs were reduced by a considerable amount.

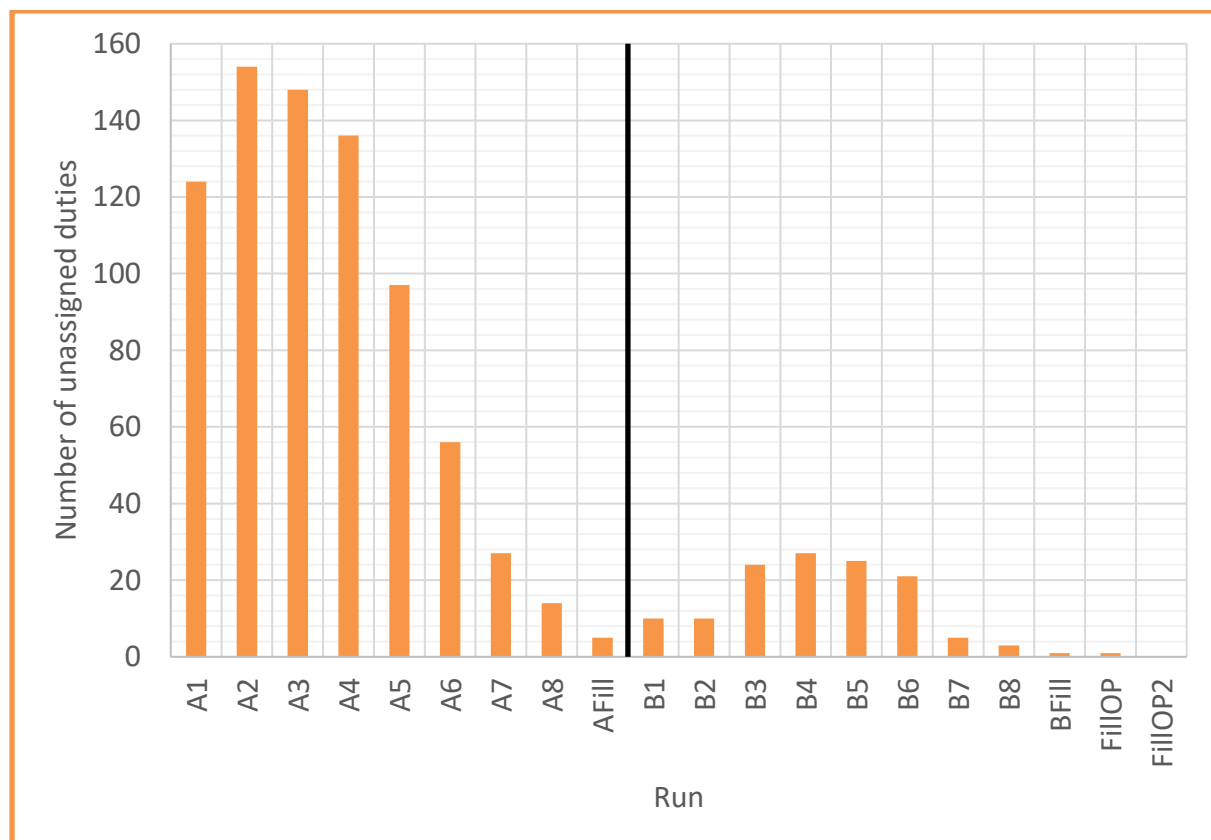
Sub-questions

What are the requirements of the aggregate workforce schedule?

First of all, schedules must be in accordance with legislation as described in the Working Hours Act (Rijksoverheid, 2010) of which a summary can be found in appendix II.I. Closely related to legislation are labor agreements defined in the CAO (De Nederlandse Veiligheidsbranche, 2017). These agreements have to be taken into account as well. Furthermore, all duties have to be covered by an employee that is suited for carrying out that duty. Last of all, total costs of the schedule should be minimized. These costs include salaries, penalties for violating preferred scheduling rules and penalties for leaving duties unassigned.

Is it necessary to use a cyclic schedule, if so what is the optimal cycle duration?

Strictly taken it is not necessary to use a cyclic schedule. However, if schedules are cyclic, it allows schedules to be structured and allows for schedules to be copied in subsequent periods. Moreover, a lot of rules as well as the salary payment are based on a 4-week period. Therefore, it is convenient to construct cyclic schedules.



Graph A, Unassigned duties

Which algorithm fits the situation of EBN?

An exact approach proved to be too complex due to the size and complexity of the problem. Therefore, a column generation scheme was applied, because of its applicability and previous successes in literature. Simply said, the concept of column generation is to start with a rough and easy to find solution and to iteratively improve this solution until no improvement is possible or until a satisfactory solution is found. This method showed satisfying results and fulfilled all objectives and thus fits the situation of EBN. *Graph A* shows the number of unassigned duties after each iteration of the developed model. It illustrates that not every iteration improves the solution, but in the long run the algorithm is able to find a solution that assigns all duties to a suitable employee and conforms to all legislation and scheduling rules.

What are appropriate objectives for an optimizing algorithm? Does the objective function consist of costs, number of employees, job satisfaction, other factors or a mix of the previous factors?

The objective for optimization is to minimize the number of unassigned duties, violations of preferred scheduling rules and costs resulting from overtime and undertime.

How can the tool be used to assist long term decision making on the tactical and/or strategic level?

The developed scheduling tool can be used to construct schedules based on different scenarios by adjusting input data and parameters without the need for major adjustments to the model. This allows management to compare schedules and its performance indicators in different scenarios and make decisions accordingly.

How does the developed tool improve the scheduling process, compared to the current scheduling process?

The developed scheduling tool allows the planning department to construct a base schedule within less time and with no penalty costs. Whereas it would take a planner multiple days to construct a base schedule, the algorithm is able to construct a base schedule in around 4 hours. Regarding the penalty costs associated with violating the preferred scheduling rules stated in chapter 2.2, the algorithm achieved great reductions as well. The current schedule of EBN contains 50 violations as shown in *Figure 6*. Considering that there are around 2000 duties in the base schedule, this means that there is a violation in 2.5% ($\frac{50}{2000} * 100 = 2.5$) of the duty assignments. The algorithm constructed a base schedule with no violations of the preferred scheduling rules.

Contents

1. Introduction.....	1
1.1. Company description.....	1
1.2. Process description	2
1.3. Scheduling.....	4
2. Problem description	6
2.1. Rules and regulations	7
2.2. Problem verification	10
2.3. Characteristics.....	12
3. Literature review.....	13
3.1. Gap.....	15
4. Problem statement.....	16
4.1. Problem definition.....	16
4.2. Research question.....	17
4.3. Sub-questions.....	17
4.4. Scope.....	18
4.5. Output.....	18
5. Methodology.....	19
5.1. Data preparation.....	19
6. Model formulation.....	21
6.1. Network flow model.....	22
6.2. Column generation scheme.....	26
6.3. Model extensions	34
7. Analysis	38
8. Conclusion	45
9. Future research.....	48
10. Bibliography	50
11. Appendix.....	52
11.1. Working Hours Act.....	52
11.2. Column generation solutions A.....	53
11.3. Column generation solutions B.....	55

I. Introduction

This research project was conducted at EBN. A change in personnel at the planning department of EBN is the reason for management to consider a new approach. The fact that new personnel is employed in the planning department allows the company to start with a clean slate. It is good to change the process of constructing aggregate workforce schedules because the schedules are currently constructed manually, which takes up a lot of time. Management at EBN would like to develop a tool that assists the planning department in constructing schedules, to reduce their workload and improve the quality of schedules.

The importance of a well-constructed workforce schedule can be deducted from the ample effort researchers have put into this topic, which is illustrated by the vast number of articles relating this topic. Van den Bergh et al. (2013) state that labor cost is a major cost component for many companies. Therefore, being able to reduce this cost by only a few percent could be very beneficial. They further explain that more and more factors are taken into account when constructing work schedules. Examples of factors to consider are different employment types (full-time, part-time and temporary) or employee preferences. This applies to EBN as well, that a reduction in workload can be financially attractive. Besides, it is important to consider the job satisfaction of employees, which can be improved by meeting more of their preferences in the constructed schedules.

Moreover, the article of Ernst et al. (2004) explains that it is critical for most companies to have the right personnel at the right time. Thus, calling for a proper work schedule in order to meet customer demand. Therefore, workforce scheduling is not only influenced by money or job satisfaction but also by meeting agreements with clients. Wan and Bard (2007) add to this by explaining that in theory it might be optimal to adjust the workforce to match demand. However, in practice this would neither be possible due to laws and contracts, nor would it be likely to lead to the desired result. Even if demand is relatively stable, adjustments have to be made to account for changes in demand and workforce availability. This further illustrates the need for a well-constructed workforce schedule and consequently the importance of this research.

I.1. Company description

EBN is a company that started as a family business in the year 1919; it was one of the first private security companies in the Netherlands. Over time, the company has expanded its business and currently offers a variety of security services in the Breda area. Examples of what they offer are locking & unlocking services, reception services, consultancy, hospitality services, site security, mobile patrols and alarm response. EBN operates in a diverse range of market sectors and clients. The company deploys guards for the government, healthcare, educational institutes, retail, business and private clients. To achieve this, EBN employs a large pool of security personnel of around 150 employees, which are good for 127 FTE, based on 152 hours per 4 weeks. On average, 93 different employees are assigned to duties every day.

1.2. Process description

In this section, the relevant processes regarding the planning department of EBN will be discussed. In addition, *Figure 1* graphically presents the discussed processes to create a clear understanding of the sequence and relationships between the processes.

Duties

EBN has agreements with every client regarding the offered services. The offered services result in one or more tasks that have to be carried out for every client. Depending on the nature of a task, a time constraint may be linked to it. For example, inspection rounds at closing time are agreed to be carried out at a certain time, whereas mobile surveillance is not carried out within a strict time frame, but rather randomly to allow for quick responses in case of alarm notifications. In order to reduce the complexity of personnel scheduling, tasks are combined as duties. One duty may consist of one or more tasks and is carried out by a single guard. In case a site requires multiple guards to be present at the same time, this is documented as multiple tasks and the site will appear in multiple duties. Constructing duties is done with the aid of a software program called RPS (Route Planning System), taking into account factors such as breaks, travel time, total duration and other constraints that may prohibit certain tasks to be paired. After the tasks are combined as duties, the planning department no longer has to consider the individual tasks and how to sequence the tasks in a schedule, when constructing the aggregate planning. The personnel scheduling problem now consists of assigning guards to all duties: a considerably less complex job, compared to a situation where all tasks are considered.

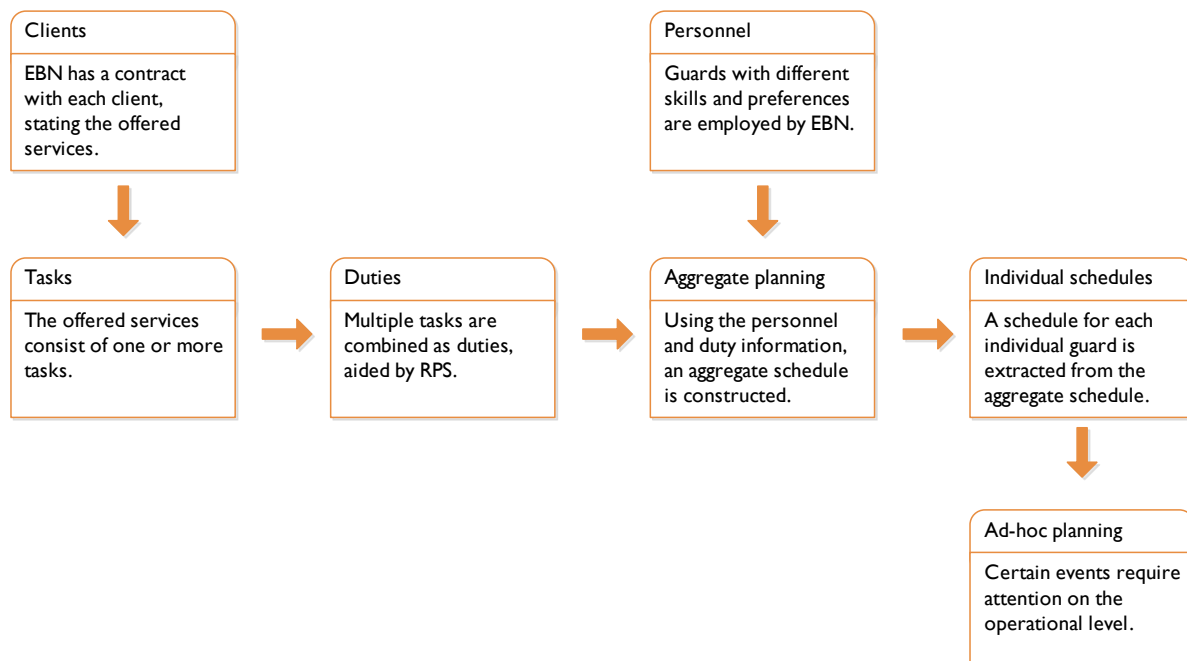


Figure 1, Process diagram of the planning department.

Aggregate planning

The next step for the planning department is to make sure that all duties are covered by a suitable guard. While a lot of the complexity is reduced by decomposing the full problem into a duty generating problem and a duty assignment problem, it is still a challenge to cover all duties because there are many constraints and many factors that influence the schedule. Important constraints come from the labor contracts (full-time, part-time or temporary), CAO (Dutch labor agreements) and legislation regarding work hours and resting times. Additionally, duties require certain skills and knowledge to be carried out properly. As a result, most guards are only deployable on a few different duties, thus restricting the flexibility of the personnel and consequently restricting the options the planning department has to find a suitable employee for every duty. Moreover, the planning department attempts to take into account employee preferences, such as working on specific days or duties.

Currently, the construction of the aggregate planning is carried out manually, even though this is a very extensive puzzle to solve. The process of scheduling is carried out with the aid of a scheduling software called CAS, indicating whether the aggregate planning is feasible. This program does not actively optimize the personnel scheduling. To be able to come up with a feasible aggregate planning in reasonable time, the planning department uses a pre-constructed base schedule. In section 1.3 is explained how the base schedule is established. In theory, the base schedule is an optimal schedule without deviations in capacity (sick leave or days off) or in demand. However, in reality there are often changes to which the schedule must be adjusted. Even though it is not the intended use of the base schedule, the now adjusted schedule is often used as a basis for the next schedule, instead of using the original base schedule. This means that less adjustments are needed to obtain a feasible aggregate planning, but the downside is that the personnel planning strays from the optimal solution.

Even though it is useful for the planning department at EBN to have an overview of all duties and guards in the aggregate planning, this is not a convenient way for guards to check their schedule. Therefore, an individual schedule is extracted from the aggregate planning for every guard. This individual schedule states the start and end time of a guard's duties, as well as which duty they must carry out, so every guard knows when and where they are expected the coming week. The process of constructing aggregate and individual schedules is further explained in section 1.3.

Ad-hoc planning

The aggregate planning aims to cover all duties, based on the information known in advance. However, several factors may disturb the day-to-day business. For example, employees calling in sick can interfere with the capacity in terms of available guards. On the other hand, additional service requests on short notice result in a sudden increase in demand. Both resulting in a need to solve gaps in the planning on an operational level. The planning department solves such issues with the aid of CAS. This scheduling program shows the planner which guards are eligible to fill in on a duty, based on their skills and availability. A problematic factor with filling gaps in the schedule is that guards are typically suited for only a few duties and a duty is only carried out by a limited number of guards. Therefore, there are limited options to replace an absentee. This sometimes results in reassigning multiple guards to different duties, which is undesirable.

1.3. Scheduling

The scheduling process consists of 3 phases: base schedule, planning phase and registration phase. A graphical representation of the 3 phases is found in *Figure 2*. The base schedule or time window planning is sent to every guard weekly, 4 weeks in advance. This schedule shows every guard in which time windows they may be scheduled. A time window has a maximum length of 10 hours per day. This makes sure that enough capacity is available to cover all duties and allows guards to plan ahead in their private day-to-day life. After the time window planning is sent, it is not allowed to make adjustments without approval of the guard in question. Next, every Thursday a duty schedule for the following week is sent, which states the duties every guard is assigned to. Since the time window planning is sent 4 weeks ahead and the actual duty planning is sent only 1 week ahead, the planning department has 3 weeks to construct a duty schedule. This phase is called the planning phase. During this period, typically more information becomes available regarding demand from clients and availability of personnel. The time window planning now represents the availability of guards. Therefore, the planning department is free to make adjustments to duties and assign duties to guards, as long as they are in accordance with the time windows. However, occasionally it is inevitable to assign a duty to a guard that differs from their time window, in this case the guard must be asked to agree first. At the end of the planning phase, the duty schedules are distributed, which marks the start of the registration phase. During this phase, it is not allowed to make adjustments without confirming with the guard in question. Even changes that are still in accordance with the earlier stated time windows, must be confirmed with the guard. This is necessary because guards are promised certain duties and work hours at this point and may have planned other activities in their private life.

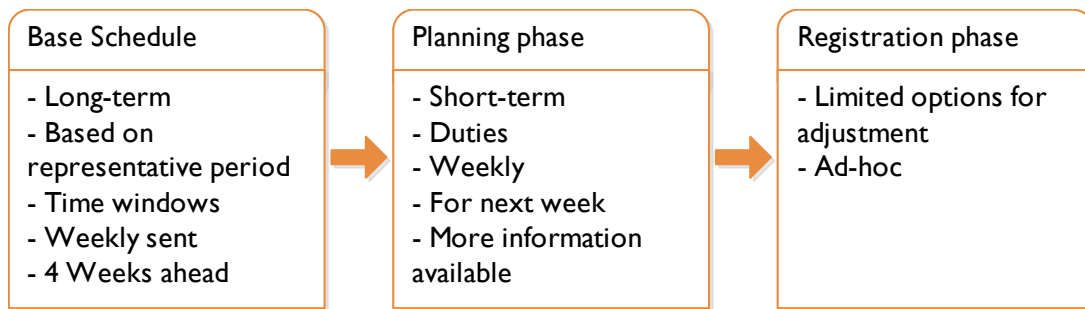


Figure 2, Three phases of scheduling

In order to reduce the workload of constructing weekly time window schedules, a base schedule is used. At the end of every year, a base schedule is constructed, that should, in principal, be applied for an entire year. Therefore, a representable period of 4 weeks is selected. Based on this representable period, a feasible time window schedule is constructed. Since the base schedule has to deal with labor agreements, the schedule is cyclic. Therefore, when a time window schedule is prepared, the appropriate week in the cycle of the base schedule is selected and adjusted to changes in demand and capacity. If such changes are expected to be structural, the changes are applied to the base schedule. This is to prevent carrying out the same adjustment weekly. However, adjustments made to the base schedule result in the base schedule becoming less and less optimal. Additionally, more gaps appear in the base schedule over time. When too many gaps exist, a new base schedule is constructed.

Since the scheduling software applied at EBN only passively supports the planning department, employees still have to assign time windows to guards manually. Passively supporting means that the software shows whether a decision results in a feasible schedule, but it does not suggest steps to achieve a proper scheduling solution. Since full-time employees are the most difficult to fit in, the planning department applies a policy of prioritizing employees based on contractual hours. This means that full-time employees are assigned first, followed by part-time employees and flex workers are assigned last.

2. Problem description

During interviews with management and the planning department, the following problems were observed.

Management does not have insight in the capacity requirements regarding the workforce. On a tactical level of decision making, the first problem here is that sometimes there is not enough capacity to fulfill all duties. This means that security guards must work overtime or that less time is spend on tasks. Working too much overtime may harm the health and satisfaction of employees. Spending less time on tasks on the other hand, could affect the quality of offered services negatively. Secondly, the lack of insight in capacity requirements, sometimes leads to overcapacity. This means that there is not enough work to meet the minimum of work hours of every guard. While this is not a problem for temporary workers, it is an issue for employees on payroll, since they have to be paid according to their minimum hours, even if they worked less. A third concern is the inability of management to predict how a change in the workforce affects the schedule. For example, during summer, it is important to know how many employees are allowed to go on holiday at the same time. Management also needs to know how many additional guards are required when accepting a new contract and consequently what the costs are. Lastly, it is a challenge to find a fitting ratio between full-time, part-time and temporary employees. The difficulty is to be able to deal with fluctuations in demand and capacity. In summary, management struggles to predict what effects a decision may have on work schedules. This is caused by the fact that schedules are constructed manually and that it takes a lot of time to construct new schedules for a new scenario. If schedules could be constructed in much less time, management would be able to compare results under different circumstances

Another problem is the lack of structure in scheduling. Because the schedules are made manually, there are a lot of deviations in work schedules, which are currently dependent on which employee made the schedule. As a result, guards have very irregular work schedules. It would be more convenient for employees to have a more constant schedule, which could positively influence the rate of absence. Additionally, it would allow for management to show a possible new employee an example of their schedule. Related to creating more structure in scheduling, is how ad-hoc planning is handled. This occurs for example when a guard calls in sick or when a client requests an additional service on short notice. Currently, solving a gap in the schedule can lead to shifting the duties of multiple guards. This is also due to the fact that guards are typically not suited for all duties. They often are employable for only a few duties, which impedes the process of replacing a guard on short notice.

In short, the identified problems are graphically presented in *Figure 3*. The problems are divided by the level of decision making they belong to. The strategic level of decision making refers to high level, long term decisions that affect the whole company. Tactical decision making occurs on a lower level and aims at meeting goals defined on the strategic level. These decisions are more short-term than the strategic decisions. Lastly, operational decision making

is the least complex and happens on a daily basis. Since this research focusses on the planning department and its related processes, the defined problems do not affect the entire company. Therefore, there are no strategic level problems stated in the figure.

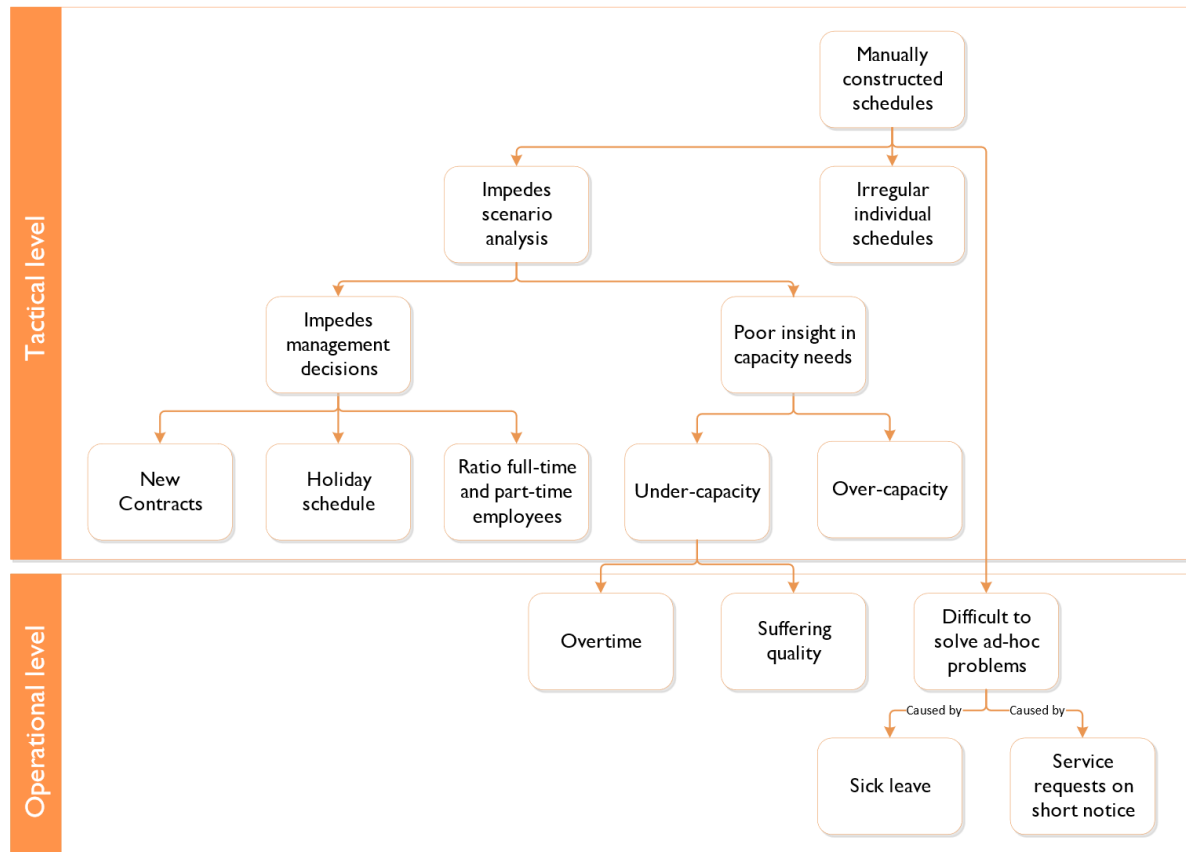


Figure 3, Problem description

2.1. Rules and regulations

Schedules for guards must be constructed according to several rules. Some rules are defined by the Working Hours Act (Rijksoverheid, 2010), while other rules are defined in the CAO (De Nederlandse Veiligheidsbranche, 2017), which are labor agreements. A summary of the legislation regarding work hours can be found in Appendix I I.1, retrieved from Rijksoverheid (2010). To get an overview of the nature of the stated problem, relevant definitions and constraints from the Working Hours Act and the CAO are listed in Figure 4 and Figure 5. More rules and regulations are defined by the Working Hours Act and the CAO, but only relevant constraints are discussed, because not all constraints apply to scheduling at EBN. For example, rules regarding breaks are taken care of during duty generating and are not relevant for the planning department. Furthermore, only the most restrictive rules are discussed. To explain: rule A applying to a 16-week period may specify a more restrictive constraint than rule B applying to a 4-week period. Since a 4-week period is considered during scheduling, the schedules must comply to both rules and only the most restrictive one is relevant.

Working Hours Act

- Maximum work time
 - (W1) 60 Hours per week.
 - (W2) 48 Hours on average per week in a 16-week period.

It is prohibited to make employees work more than 60 hours in a single week. Furthermore, employees are allowed to work 48 hours on average per week in a 16-week period. When scheduling for a 4-week period, the same 48 hours weekly average should be applied to comply with the 16-week average.

- Rest times
 - (W3) Daily rest: 11 hours rest after every duty.
 - (W4) Weekly rest: 36 hours consecutive.

After a working day, an employee must have 11 consecutive hours of non-work time. In the event of a 5-day work week, an employee must have 36 consecutive hours of non-work time after the end of the work week.

- (W5) 13 Free Sundays per 52-week period.

An employee must have at least 13 free Sundays per year. Since a period of 4 weeks is considered, this translates to at least one free Sunday per 4-week period: $52/13=4$.

- Night duties
 - (W6) At most 40 hours per week if ≥ 16 night duties per 16 weeks.
 - (W7) At least 14 hours rest time after a night duty that ends after 02:00h.
 - (W8) At least 11 hours rest time after a night duty that ends before 02:00h.
 - (W9) At least 46 hours rest time after a series of duties containing at least 3 night duties.
 - (W10) At most 36 night duties per 16 weeks.

An employee works a night duty if they work more than 1 hour between 00:00h and 06:00h. Stricter regulations apply for night duties than for day duties. If an employee works more than 16 night duties per 16 weeks, they may work at most 40 hours per week. Additionally, at least 14 hours of non-work time must follow after a night duty that ends after 02:00h. After a series of duties containing at least 3 night duties, an employee must have at least 46 hours of non-work time after the last night duty. Last of all, an employee is allowed to work at most 36 night duties per 16 weeks. This means at most 9 night duties if a 4-week planning horizon is considered: $36/4=9$.

Figure 4, Working Hours Act

CAO

Definitions:

- Day off: Day without a starting time between 00:00h and 24:00h, 24 hours without work.
- Day duty: Starts after 06:00h and ends before 20:00h.
- Evening duty: Ends after 20:00h and before 2:00h.
- Full-time contract: 152 Hours per 4 weeks.
- Night duty: More than 1 hour work between 00:00h and 06:00h.
- Overtime: Hours exceeding 152 hours per 4 weeks. If employees work more hours than their contract, this does not necessarily result in overtime.
- Part-time contract: Less than 152 hours per 4 weeks.
- Undertime: Hours less than contract hours.
- Weekend: Period of 2 consecutive days off, latest start after evening duty on Friday and ends after 05:30h on Monday.

Constraints:

- (C1) At most 20 duties per 4-week period
- (C2) At most 160 hours of work time per 4-week period.

This rule is more restrictive than the rule in the Working Hours Act (WHA) allowing at most 48 hours on average in a 16-week period: $4 \times 48 = 192$ hours > 160 hours. Therefore, a maximum of 160 hours per 4-week period is considered when constructing schedules.

- (C3) At most 32 night duties per 13-week period.

The WHA is more restrictive on the number of night duties, because it allows only 9 night duties per 4 weeks as opposed to 32 night duties per 13 weeks. 32 Night duties per 13 weeks is equal to 9.85 night duties per 4 weeks.

- (C4) At least 2 periods of 2 consecutive days off per 4 weeks. One of these periods should cover the weekend.
- (C5) At least 1 free Sunday per 4-week period.

If an employee has 2 consecutive days off that cover the weekend: Saturday and Sunday, they consequently have a free Sunday. Therefore, the rule of a free Sunday is not considered as a separate rule.

- (C6) At most 7 consecutive night duties.
- (C7) At least 48 hours rest after a series containing at least 3 night duties.

The CAO is more restrictive than the WHA that requires only 46 hours. A series of night duties starts after a weekly rest (at least 36 hours) and ends with the next weekly rest.

Figure 5, CAO

2.2. Problem verification

This section presents data from EBN with the goal to support the statements made above regarding the existing problems. It is not the goal to give a detailed description of the situation with the data presented below, but rather to show that there is in fact a problem and that there is room for improvement.

Preferred scheduling rules

Of course, the schedule of EBN has to comply with government regulations. Beside that they want proper schedules for their employees with sufficient rest times and appropriate duty combinations. Therefore, they have come up with preferred scheduling rules. Schedules that disregard these preferred scheduling rules still follow the minimum requirements as defined by the CAO and Working Hours Act but are less desirable from the view of EBN and its personnel. The objective is to follow these scheduling rules as much as possible by penalizing less desirable schedules. Schedules generated by this research project will be compared with the ones generated by EBN based on these preference penalties. The table in *Figure 6* depicts violations of the preferred scheduling rules over a period of 4 weeks in 2018. Every column shows the number of observed violations during these 4 weeks for the corresponding rules. An explanation of the rules is given below *Figure 6*. It is the aim of EBN to provide schedules to its personnel that are as favorable as possible. As can be seen in *Figure 6*, there is room for improvement, since there have been several violations. Values of the preferred scheduling rules are not given for confidentiality reasons.

Violation	Rest time after ≥ 3 night duties	Weekly rest	Daily rest
Frequency	22	12	16

Figure 6, Frequency of violations of preferred scheduling rules.

Rest time after ≥ 3 night duties: Employees must have at least 48 hours of rest after a series of duties that contains at least 3 night duties. If the last night duty ends on Tuesday morning at 6 am, for instance, the employee may not resume work until Thursday at 6 am. EBN prefers a rest of X_1 (≥ 48) hours after a series of duties containing at least 3 night duties.

Weekly rest: Employees must have 36 consecutive hours of rest every period of 7 times 24 hours. EBN prefers X_2 (≥ 36) consecutive hours of rest every period of 7 times 24 hours.

Daily rest: Employee must have 11 hours rest after a day duty. EBN prefers X_3 (≥ 11) hours of rest after a day duty.

Distribution of skills

In this case, a skill refers to the ability of an employee to carry out a certain duty and only one skill is required per duty. The number of skills per employees can be found in *Figure 7* and should be interpreted as follows: a value of 24 on the y-axis for a value of 3 on the x-axis means that there are 24 employees with 3 different skills. Furthermore, there are no guards

that can be assigned to exactly 24 different duties and 1 employee is suited for 27 different duties. The table in *Figure 7* shows a broader spread of skills than was expected; there are ample employees with more than five skills. However, a large part of the employees can only be assigned to one or two duties. In case it shows that flexibility among personnel is an issue, the cause may be found in the number of skills per employee and this could possibly be solved by cross-training. A similar conclusion is drawn from *Figure 8*, which depicts the number of employees per skill. It reveals that a number of duties can only be fulfilled by one or two guards. This may be problematic when one of those guards calls in sick or requests a day off. Opposed to having an employee with only one skill, a skill with only one employee is more difficult to deal with, since there is no one to replace the guard.

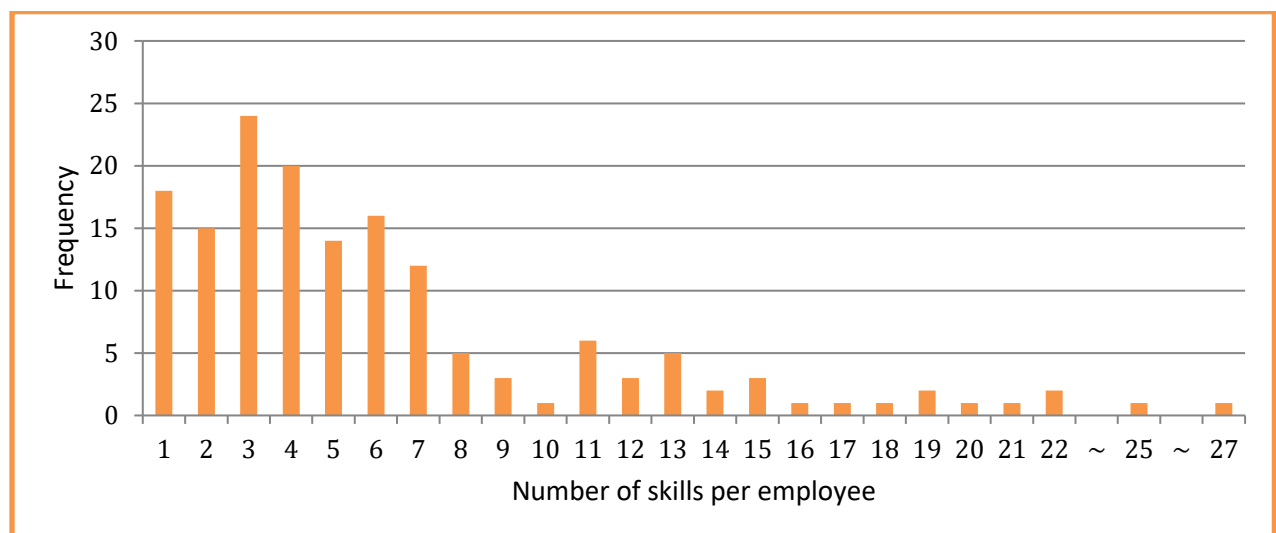


Figure 7, Frequency of the number of skills per employee



Figure 8, Frequency of the number of employees per skill

2.3. Characteristics

In order to find appropriate literature and a suitable solution method, the problem is specified according to the taxonomy provided by Ernst et al. (2004). This article divides the scheduling process into several modules. Within a module, different models may be needed for specific applications (Ernst et al., 2004). In this section, it is explained for all relevant modules, which model fits best to depict the problem at EBN.

Demand modelling: In the case of EBN, tasks are already combined as duties before the schedule is constructed. Furthermore, demand is rather stable and known in advance. Therefore, shift-based demand is deemed to be the best description of the situation.

Days off scheduling: This module has to be considered as well, since the length of a work week of guards does not equal the operational week of the company.

Line of work construction: Currently, line of work is constructed using a cyclic schedule, rotating after 4 weeks. This makes sense, because demand shows repeating patterns. Additionally, regulations restrict the number of consecutive night-duties. By adopting a cyclic schedule, night- duties can be divided over guards fairly, while following labor legislations. Moreover, salaries are paid every 4 weeks. Since the compensation for a week consisting of night duties may be different from a regular week, it is convenient to rotate the schedule with the salary cycle.

Staff assignment: Duties are already defined, so at first sight it is the assignment of guards to duties that remains. However, looking further into the situation at EBN, it is revealed that the planning department has to deal with time windows as well. Section 1.3 defined time windows and explains how they are applied. The characteristic of time windows being present, makes this problem interesting in the scientific field of operations management, because a decision must be made on how to assign time windows. In contrast to a more regular scheduling problem, which relates more to the mathematical field, because it mainly involves modelling a problem instead of making decisions regarding the practical application.

Based on the characteristics described above, it can be concluded that this research has a lot in common with scheduling problems in healthcare. This is due to the fact that both deal with a scheduling problem where:

- Capacity needs are mostly known beforehand. In healthcare it is predetermined how many nurses have to be scheduled.
- The length of a work week does not equal the length of an operational week.
- Line of work construction involves cyclic schedules.
- Staff assignment is the main issue, due to the different skills required for different duties.

3. Literature review

This section presents a brief literature review on the topic of scheduling problems. To get an overview of the taxonomy and different approaches in scheduling problems, the literature reviews by Ernst et al. (2004) as discussed in section 2.3. and Van den Bergh et al. (2013) are used as a basis. The first article gives a framework to categorize scheduling problems based on their characteristics along with examples of application areas. The second provides an overview of previous literature categorized by the problem setting or technical features of the scheduling problem discussed in the articles.

In literature, mainly three approaches are presented to solve scheduling problems. Naudin et al. (2012) briefly summarize the different methods. The first approach is the exact approach, which involves the use of mathematical models based on the set covering problem. Sometimes the size or the characteristics of a problem call for different methods. A second approach is decomposition, this method divides the problem in two or more sub-problems in order to reduce the complexity of the problem. Thirdly, heuristics are applied to solve scheduling problems.

Exact approach

In contrast to heuristic algorithms, integer programming guarantees optimality. However, it is a time consuming and difficult task to implement a good integer programming method or any other exact method, for a particular scheduling problem. This is only justified when the reduced costs as a result from the solution are significant and when the scheduling rules and regulations are relatively static over time (Ernst et al., 2004). Heuristics on the other hand, are not guaranteed to produce an optimal solution. However, they tend to be relatively robust and can deal with a wide range of input data. Additionally, heuristics can usually produce a good feasible solution in a short amount of time, as stated by Ernst et al. (2004).

Decomposition

Large problems can be decomposed into sub-problems. This helps to reduce the complexity of the scheduling problem. Ernst et al. (2004) give the example that it makes sense to deal with rules governing work breaks while setting up pairings rather than including them as a set of possibly complex side constraints in an optimization model. They further add that the complexity associated with detailed work rules may be difficult to integrate in a full optimization model, but often has only limited effect on the overall solution quality. One decomposition approach is called column generation, where for example, possible schedules (columns) are generated by a separate algorithm. Subsequently, the master problem consists of selecting the best columns for the solution. Several methods exist in column generation approaches regarding the generating of columns. They may be generated dynamically, exhaustively or by a heuristic. Examples of situations where column generation has been applied to tackle scheduling problems in healthcare are found in Brucker et al. (2005) and Bard and Purnomo (2005). Furthermore, an application in scheduling train drivers and guards

is found in Kroon and Fischetti (2000) and Wan and Bard (2007) describe an application in a production facility. One of the papers that studied decomposition for workforce assignments in the recent literature is Firat et al. (2016) where they propose a Branch-and-Price algorithm for stable workforce assignments with hierarchical skills.

Whereas the above presented researches show the value of decomposing a complex problem, Maenhout and Vanhoucke (2013) propose to incorporate as much decisions as possible in one model. They argue that isolated reasoning leads to suboptimal decisions and try to overcome these inefficiencies by proposing an integrative nurse staffing and shift scheduling approach. Maenhout and Vanhoucke (2013) further present their findings regarding different policy decisions on the staffing level. These include the ratio between full-time, part-time and temporary workers as well as the effects of cross-training of personnel.

Heuristics

Heuristics can be used to solve very complex problems but do not guarantee that the solution is optimal. Marasco and Romano (2011) further explain the need for heuristic algorithms or approximation procedures by stating that almost all scheduling problems are NP-hard. This means that in many situations it is currently not possible to find an optimal solution within reasonable time. The article explains that this difficulty pushed researchers to develop heuristic algorithms. Firat and Hurkens (2012) show how a heuristic can be helpful to find good quality solutions to a complex problem in a short amount of time. They developed a MIP-based approach that achieved significant improvement in the constructed schedules for a technician task scheduling problem. In this problem tasks require skills and teams are to be formed while combining the tasks as their workloads.

Mathematical models

Formulating a mathematical model to describe a scheduling problem can be done in many ways and has to be customized to the case. This means that it is not possible to construct a general model that can be applied in every scheduling problem. However, some aspects of scheduling appear in many cases. Brucker et al. (2011) present a general formulation of common scheduling problems, which can be used to describe general scheduling cases, or as a basis for a more specific case. Many extensions of these general formulations can be thought of. One of those extensions is a situation where work is done around the clock and 7 days a week. In comparison to a workweek of 5 days from nine to five, it is much more complex to schedule personnel for 7 days a week. A reason for this is that this situation requires scheduling days off and resting hours. To overcome the difficulty of scheduling personnel in compliance with labor legislation and agreements, it is common to apply cyclical schedules. A description of work cycles, how they can be used and a comparison with non-cyclical schedules is given by Chan and Weil (2001). Furthermore, they present mathematical models to construct cyclical schedules using constraint programming.

3.1. Gap

One of the trends observed in literature, is that the solutions to scheduling problems are typically focused on specific aspects of scheduling. Integrating all decisions related to scheduling is not a common approach. Instead, many problems are decomposed in order to reduce the complexity of the problem. Van den Bergh et al. (2013) and Ernst et al. (2004) make the same observation and mark this area as a major research opportunity. The benefit of integrating all decisions in a single model is supported by Maenhout and Vanhoucke (2013) in their research to construct an integrative nurse staffing and shift scheduling model.

An important aspect of scheduling is to be prepared for future events. It is observed though, that researchers often apply a rather deterministic approach. It is assumed in those cases that demand and capacity are known and irregularities such as sick leave or unforeseen service requests are not considered. Van den Bergh et al. (2013) propose to incorporate uncertainty in the decision-making process, or to test the robustness of the model by running simulations.

Not considering all relevant decisions in a model as well as disregarding uncertainty, make a model less effective in describing reality. A model that does not perform properly in practice is not useful to implement in a business environment. The same conclusion is drawn by Van den Bergh et al. (2013). The article states that many researches do not make it to implementation, because they lack integrating different scheduling problems as well as the low degree of uncertainty incorporation.

4. Problem statement

In this section, the problem statement is defined in five parts. First, a definition of used concepts is given to give the reader an understanding of the structure of the problem. Next, the research question is determined, which is the main guideline of the research project. As a result of the steps taken in this research, an answer to the research question is presented in the conclusion of this paper. Thirdly, several sub-questions are stated, that help answering the main research question in a structured fashion. Next, the scope of this research is defined. The scope ensures that the research has enough depth, at the same time it limits the project. and prevents it from exceeding the time limit. Finally, a list of outputs is stated, which shows what must be included in the results of this research.

4.1. Problem definition

Period. The smallest unit of time considered in the scheduling problem. The number of periods in a model depends on the planning horizon and granularity of the model. A model with a granularity of 1 hour and a 4-week planning horizon would result in $24 \times 7 \times 4 = 672$ periods to cover all periods of time in the model.

Employee. Employees are regarded as renewable resources; whose availabilities are given by their contract types and have skills to perform the duties.

Skill. Skills link employees to duties. If an employee has the skill for a duty, it means that they are suited to carry out that duty. An employee may have multiple skills and duties may be linked to multiple employees.

Duty. Combination of one or more tasks which can be carried out by a single employee in one day. Duties are linked to employees through skills as explained above and are linked to periods by a start and end time. Duties can start and end at any moment of a day. Constructed duties conform to legislation regarding maximum work time, breaks and other rules that apply to a single duty.

Rest. Periods of non-work time. Rules defined by law and labor agreements state that every employee must have at least certain periods of rest. Depending on what rule is considered, a rest period may or may not start at every moment. For example, days off must start at 00:00h, while a weekly rest of 36 hours may start at any given moment.

Individual schedules. A combination of zero or more duties that considers legislation and other rules. In this project a schedule covers the entire planning horizon of four weeks. Every employee is linked to exactly one individual schedule stating what duties they are assigned to and consequently when and where they have to work and when they are free.

Aggregate schedule. The collection of all individual schedules. The aggregate schedule shows for every employee to what duties they are assigned and consequently it shows for every duty which employee will carry it out. This research considers only the regular duties of EBN. Regular duties are defined in contracts with clients and have to be carried out for a longer period, often for at least one year.

To give a better understanding of duties, examples and further explanation will be given. First of all, as explained in section 1.2 duties are a combination of tasks that can be carried out by a single employee in one day. Duties take into account all relevant restrictions such as travel time between tasks, breaks, maximum worktime per duty and so on. This means that the planning department does not have to consider rules regarding individual duties and only have to consider rules such as rest between duties, weekly rest and maximum overtime. A duty is defined by a start time, end time, weekday and the tasks that it consists of. An example of a task that EBN carries out for its clients is 24/7 site security. This task cannot be carried out by a single employee and is therefore split into three 8-hour duties per day with fixed start and end times. If the client requires 2 guards on 24/7 site security, this would result in six separate duties per day to allow for duties to be carried out by a single employee. Another duty may consist of a mobile patrol where an employee patrols for example an industrial area from 22:00h to 07:00h. A duty could also consist of multiple tasks, such as locking and unlocking services where an employee locks or unlocks a building, checks for irregularities and goes on to a next task.

4.2. Research question

How can the scheduling process at EBN be improved by developing and implementing a scheduling tool that applies an optimization algorithm?

4.3. Sub-questions

Tactical and strategic level

- What are the requirements of the aggregate workforce schedule?
- Is it necessary to use a cyclic schedule, if so what is the optimal cycle duration?
- What is the structure of duties? Does it allow flexibility in the work schedule, or does it force certain work sequences?
- How to deal with time windows? Is it necessary to schedule time windows in such a fashion that the assigned duties never violate constraints?
- Which algorithm fits the situation of EBN?
- What are appropriate objectives for an optimizing algorithm? Does the objective function consist of costs, number of employees, job satisfaction, other factors or a mix of the previous factors?
- Which requirements and objectives should be modelled as hard constraints, and which should be modelled as soft constraints?
- How can the tool be used to assist long term decision making on the tactical and/or strategic level?

Operational level

- How does the developed tool improve the scheduling process, compared to the current scheduling process?

4.4. Scope

The most important part of this research is to solve the problem of the planning department regarding the construction of a base workforce schedule. As stated in chapter 2.1, this is currently done by hand and leaves a lot of room for improvement. Due to the characteristics of the duties and labor agreements, it is most efficient to apply a cyclic schedule. A cyclic schedule can overcome issues related to rest days after night duties and switching between duties. Additionally, a cyclic schedule allows for the workforce to have a regular schedule.

A scheduling tool can be very helpful for long term scheduling. It can give management insight in how to deal with more expected variations in demand and capacity. By adjusting the input for the model, to depict the expected change, the decision maker can see the results of such a scenario and act accordingly. For example, during the holiday season, a scenario analysis can reveal how many guards can be allowed to take vacation safely, without ending up with too little capacity. In a similar fashion, it allows management to see whether it is possible to take on a new client with the current workforce.

Outside scope

From the point of view of this research project, generating duties is a preliminary step. This step involves combining tasks as duties that can be carried out by a single guard and have to follow certain constraints. It reduces the complexity of the planning process drastically to decompose the problem into a duty generating problem and a duty assignment problem. Furthermore, it is expected that the duty generating is already done efficiently and that an integration would yield minor benefit. Currently, the planning department is not involved in generating duties and it would fit the situation at EBN to limit this project to the duty assignment problem. Therefore, duty generating is left outside the scope of this research.

4.5. Output

Together with the management of EBN, a list of requirements was formulated. This list is presented below and was used to prevent a mismatch between the results of this research and the expectations of the managers.

- A scheduling tool that automatically generates a base aggregate workforce planning, using input regarding expected demand and capacity, while considering labor agreements.
- The obtained workforce planning must be cyclic.
- The tool must help management to gain insight in capacity needs.
- The tool must be tested and implemented at the planning department.

5. Methodology

As described in chapter 3, many approaches have been carried out by researchers to solve scheduling problems. The main categories of approaches are exact, heuristics and decomposition approaches. From these categories, it would be ideal if an exact approach can be carried out because this ensures that the formulated model is solved to optimality. Proving the quality of the solution would therefore be unnecessary in this case. However, some problem instances are too complex to formulate in a linear fashion, which is required for exact algorithms. Those problems can only be solved using heuristics. Even though these approaches can deliver an excellent solution, with a marginal gap to optimality, it remains a challenge to prove the quality of the solution. The quality is often demonstrated by comparing its results to other heuristics or to the results prior to the integration of the solution. Additionally, applying a heuristic requires the programming of a customized algorithm, whereas when an exact approach is applied, the formulated model can be solved by of-the-shelf algorithms. Last of all, a problem may follow the linearity requirement for an exact solution but can simply be too large to solve in an acceptable amount of time. In that case, there is the option to decompose the problem into less complex sub-problems. This allows the algorithm to find solutions much quicker, without losing too much of the solution quality. It is in some cases even possible to maintain the guarantee of an optimal solution, by applying column generation.

To the best of my knowledge, it is not possible to determine the complexity of an optimization problem before modelling it. Therefore, it was first attempted to formulate a mathematical model that can be solved to optimality using an exact approach. However, it showed that it is not possible to solve this model to optimality within reasonable time. The next step was to reduce the model's complexity by using column generation.

5.1. Data preparation

Formulating a mathematical model that depicts reality as good as possible requires information regarding the inputs, outputs, requirements and constraints of the model. A short explanation is given for all objects, as well as a description of how the data was obtained.

Inputs

- Personnel information. This should state the agreements EBN has with every employee. This includes the number of hours every guard has to or is allowed to work per day, week and month. Additionally, it is necessary to know which skills every guard has, or in other words which tasks or duties they are suited for.

This information was obtained from the planning department and is digitally stored in their scheduling software CAS. This data was exported to an excel spreadsheet and further transformed into usable data.

- Duty information. Contains the start- and end-time of every duty.
This information was obtained from the planning department and is digitally stored in their scheduling software CAS. This data was exported to an excel spreadsheet and further transformed into usable data.

Outputs

- Requirements of the workforce schedule. Aside from legal constraints and agreements with personnel, objectives and other restrictions can be modelled. Objectives can be based on performance indicators or can be based on preferences, such as the overall fairness of the resulting schedules.
These are requirements that cannot be found in existing data but were obtained by discussing these matters with management. Interviews with management revealed the requirements of the resulting schedule.

Constraints

- Labor agreements. General rules are in place regarding work conditions of guards. These include rules regarding days off, the maximum number of consecutive work days and other rules that restrict how an employee can be scheduled. As the name states, these are agreements and they may be disregarded if a guard agrees.
This information can be found in a document stating the labor agreements, which can be found on the website of the security branch (De Nederlandse Veiligheidsbranche, 2017).
- Labor legislation. Some restrictions are stated by law. For example, the maximum number of hours an employee is allowed to work per day and the amount of rest an employee needs between two duties. Restrictions coming from labor legislation cannot be violated, since this would mean a violation of the law.
This information can be found in a document stating the labor legislation, which can be found on the website of the Dutch government (Rijksoverheid, 2010).

6. Model formulation

In this chapter, the mathematical models used to construct schedules are presented and discussed. Chapter 6.1 presents a network flow model that can be used to solve the scheduling problem to optimality. The concept of a network flow model is a specific application of linear programming where all duties, rests and idle times are represented by arcs. Constraints of the model ensure that for every employee a sequence of arcs is chosen in such manner that an employee is assigned to no more than one arc at a time and that all rules are taken into account. Network flow models are a convenient way to model capacity and personnel planning and are often used for this purpose. Prior to modelling the problem as a network flow model, this project started by modelling the problem as a mixed integer set covering model. Overlapping duties in this preliminary model were prevented by defining sets of conflicting duties and could thus not be carried on subsequent days. Defining these sets was a time-consuming task and the structure of the model proved to be inconvenient to define more complex rules. Therefore, it was decided that a network flow modelling was a better approach to the problem.

During research the network flow model also showed to be too large to solve to optimality due to the vast number of variables and constraints. With 150 employees and 2000 separate duties to be linked to each other while taking into account all constraints, the problem becomes too large and complex to solve within reasonable time on a desktop PC. After running the model for 8 hours, a feasible solution was still not found.

Since solving the direct Integer Linear Programming formulation is not efficient for the computational time, the problem was decomposed by a column generation scheme with the aim to reduce complexity and to find a satisfying solution within reasonable time. The column generation scheme is described and explained in detail in chapter 6.2. Results of the calculations and a comparison with the current situation are presented in chapter 7.

6.1. Network flow model

This chapter will present the network flow model that was constructed in this research. A network model is a specific type of modelling used to solve linear programming problems. In this model, periods of time are represented by so-called nodes which are connected by directed arcs. These arcs represent either duties, rests or idle times and are further defined by a start and end time. The first node in the model represents the start of the planning horizon and is called the source node. Similarly, the last node in the model represents the end of the planning horizon. The goal, in general, is to find a path for every employee that starts at the source node and ends at the sink node. However, in this research a cyclic schedule is considered. Therefore, there is no source and sink node. Instead, every path must have exactly one arc that either starts or covers the first period. Furthermore, a path must consist of a set of consecutive arcs, meaning that arcs cannot overlap and that an arc can only start where another arc ends.

In this application, it is the objective to find a feasible path from start to end. Therefore, this network model is specifically a network *flow* model. A feasible path is a combination of arcs that conforms to all constraints and represents an individual schedule that conforms to all rules and regulations. Closely related to network flow models and the scheduling problem at EBN are shortest path problems. A shortest path model tries to find a path from source to sink with minimum costs. However, the model presented below considers the entire aggregate schedule at once which contains all employees and duties. For every employee there exists one path, or individual schedule, that together make up the aggregate schedule. Therefore, it is not possible to determine the costs of separate arcs and model the problem as a shortest path problem.

Sets

- P** **Periods.** This model has a set of $\mathbf{P} = \{0, \dots, 671\}$ periods, indexed by p . Every period $p \in \mathbf{P}$ refers to a time period of 1 hour. This model considers a planning horizon of 4 weeks. Therefore, there are $24 \cdot 7 \cdot 4 = 672$ periods in this model.
- E** **Employees.** Indexed by e .
- A** **Arcs.** Arcs connect two time periods and can either represent duties, rests or idle arcs. Indexed by a .
- $\mathbf{A}^{\text{Cycle}}$ **Cycle Arcs.** Arcs starting at or covering period $p = 0$, subset of \mathbf{A} .
- \mathbf{A}^{Duty} **Duties.** Arcs related to duties, subset of \mathbf{A} . Duties are a combination of one or more tasks which can be carried out by a single employee in one day. Duties in this model are linked to arcs and rest time between duties is included in every arc.
- $\mathbf{A}^{\text{Night}}$ **Night duties.** Arcs related to night duties, subset of \mathbf{A}^{Duty} .
- \mathbf{A}^{Rest} **Long rests.** Arcs related to long rests, subset of \mathbf{A} . A long rest is a timeframe during which an employee is given $X_2 (\geq 36)$ or $X_1 (\geq 48)$ hours rest and cannot be assigned to duties. An $X_3 (\geq 11)$ hour rest is already included in every duty-arc. For that reason the long rest arcs span a timeframe of X_4 hours ($X_2 - X_3 = X_4$) and X_5 hours ($X_1 - X_3 = X_5$) respectively.

- A^{48}** **48-Hour rests.** Arcs related to rests of 48 hours starting at 0:00h, subset of A^{Rest}
- $A^{Weekend}$** **Weekend.** Rests of 48 hours covering a weekend, subset of A^{48}
- W** **Weeks.** This model considers a planning horizon of 4 weeks. Set $W = \{1, \dots, 4\}$ is indexed by w .
- D** **Days.** The planning horizon consist of 28 days. Set $D = \{1, \dots, 28\}$ is indexed by d .
- K** **Preference penalty.** An arc may be assigned to an employee according to minimum requirements instead of following preference rules. In that case the end time of the arc changes to account for the reduction in rest time. Set $K = \{0, 1\}$ is indexed by k which represents the penalties on preference rules. Index $k = 1$ if the arc is carried out disregarding the preference rules, $k = 0$ in all other cases.

Parameters

$s_{a,e}$	= 1	If employee $e \in E$ is skilled for duty $a \in A^{Duty}$
	= 0	Otherwise
$start_{a,p}$	= 1	If arc $a \in A$ starts at period $p \in P$
	= 0	Otherwise
$end_{a,p,k}$	= 1	If arc $a \in A$ ends at period $p \in P$ for preference penalty indicator $k \in K$
	= 0	Otherwise
n_a	= 1	If duty $a \in A^{Duty}$ is a night duty
	= 0	Otherwise
h_a		Workhours related to $a \in A^{Duty}$
fte	= 152	Number of workhours over a 4 week period for a full-time employee
c_e		Minimum number of hours for employee $e \in E$
$maxWH$	= 60	Maximum number of workhours per week per employee
$maxD$	= 20	Maximum number of duties per 4 weeks per employee
$maxOT$	= 8	Maximum amount of overtime per employee, in hours.
$maxN$	= 9	Maximum number of night duties per 4 weeks per employee
$min48$	= 2	Minimum number of 48-hour rest periods per 4 weeks
$r48_a$	= 1	If arc $a \in A$ is a rest of 48 hours
	= 0	Otherwise
$aw_{a,w}$	= 1	If arc $a \in A$ starts in week $w \in W$
	= 0	Otherwise
$ad_{a,d}$	= 1	If arc $a \in A^{Rest}$ starts at most 6 days before day $d \in D$
	= 1	If arc $a \in A^{Duty}$ starts at most 2 days before or 4 days after day $d \in D$
	= 0	Otherwise
q_i		Penalty coefficient for factor $i \in \{1, \dots, 4\}$ in the objective function
M	= $1e10$	A large number

Variables

$Y_{e,a,k}$	= 1 If employee $e \in \mathbf{E}$ is assigned to arc $a \in \mathbf{A}$ with preference penalty indicator $k \in \mathbf{K}$.
	= 0 Otherwise
U_a	= 1 If duty $a \in \mathbf{A}^{Duty}$ is not assigned to an employee
	= 0 Otherwise
OT_e	Total amount of overtime for employee $e \in \mathbf{E}$, in hours
UT_e	Total amount of undertime for employee $e \in \mathbf{E}$, in hours
TH_e	Total amount of workhours for employee $e \in \mathbf{E}$
$WH_{e,w}$	Amount of workhours for employee $e \in \mathbf{E}$ during week $w \in \mathbf{W}$

Minimize

$$\sum_e [q_1 * OT_e + q_2 * UT_e] + \sum_{a \in \mathbf{A}^{Duty}} [q_3 * U_a] + \sum_{e,a,k:k=1} [q_4 * Y_{e,a,k}]$$

Subject to

- (1) $\sum_{e,k} [Y_{e,a,k} * s_{e,a}] = 1 - U_a \quad \forall a \in \mathbf{A}^{Duty}$
- (2) $\sum_{a,k} [Y_{e,a,k} * start_{a,p} * s_{e,a}] = \sum_{a,k} [Y_{e,a,k} * end_{a,p,k} * s_{e,a}] \quad \forall e \in \mathbf{E}, p \in \mathbf{P}$
- (3) $\sum_{a \in \mathbf{A}^{Cycle},k} [Y_{e,a,k}] \leq 1 \quad \forall e \in \mathbf{E}$
- (4) $\sum_{a \in \mathbf{A}^{Rest},k} [Y_{e,a,k} * ad_{a,d}] \geq 1 \quad \forall e \in \mathbf{E}, d \in \mathbf{D}$
- (5) $\sum_{a \in \mathbf{A}^{Duty},k} [Y_{e,a,k} * s_{e,a} * h_a] = TH_e \quad \forall e \in \mathbf{E}$
- (6) $\sum_{a \in \mathbf{A}^{Duty},k} [Y_{e,a,k} * s_{e,a} * h_a * aw_{a,w}] = WH_{e,w} \quad \forall e \in \mathbf{E}, w \in \mathbf{W}$
- (7) $\sum_{a \in \mathbf{A}^{48},k} [Y_{e,a,k}] \geq min48 \quad \forall e \in \mathbf{E}$
- (8) $\sum_{a \in \mathbf{A}^{Weekend},k} [Y_{e,a,k}] \geq 1 \quad \forall e \in \mathbf{E}$
- (9) $\sum_{a,k} [Y_{e,a,k} * ad_{a,d} * (n_a - M * r48_a)] - 2 \leq 0 \quad \forall e \in \mathbf{E}, d \in \mathbf{D}$
- (10) $\sum_{a \in \mathbf{A}^{Duty},k} [Y_{e,a,k} * ad_{a,d} * (n_a + 3 * (1 - n_a))] \leq 5 \quad \forall e \in \mathbf{E}, d \in \mathbf{D}$
- (11) $\sum_{a \in \mathbf{A}^{Night},k} [Y_{e,a,k}] \leq maxN \quad \forall e \in \mathbf{E}$
- (12) $TH_e \geq c_e - UT_e \quad \forall e \in \mathbf{E}$
- (13) $TH_e \leq fte + OT_e \quad \forall e \in \mathbf{E}$
- (14) $OT_e \leq maxOT \quad \forall e \in \mathbf{E}$
- (15) $WH_{e,w} \leq maxWH \quad \forall e \in \mathbf{E}, w \in \mathbf{W}$
- (16) $\sum_{a \in \mathbf{A}^{Duty},k} [Y_{e,a,k}] \leq maxD \quad \forall e \in \mathbf{E}$
- (17) $Y_{e,a,k} \in \{0,1\} \quad \forall e \in \mathbf{E}, a \in \mathbf{A}$
- (18) $OT_e, UT_e, TH_e, WH_{e,w} \geq 0 \quad \forall e \in \mathbf{E}, w \in \mathbf{W}$

- (1) **Coverage:** Ensures all duties are carried out by exactly one employee suited for the duty.
- (2) **Flow:** For every period p , the number of outgoing arcs must be equal to the number of incoming arcs for every employee e .
- (3) **Cycle:** Ensures that there is at most one sequence of arcs for every employee, comparable to a source and sink node in non-cyclic network models.
- (4) **Rest period:** Ensures that every employee e has at least one rest period of 36 consecutive hours for every period of 7×24 hours: constraint W4 in Figure 4.
- (5) **Total workhours:** Stores for every employee e their total worktime in variable TH_e .
- (6) **Weekly workhours:** Stores for every employee e their weekly worktime for week w in variable $WH_{e,w}$.
- (7) **48-hour rest:** Every employee e has at least the minimum number of 48-hour rests: constraint C4 in Figure 5.
- (8) **Weekend rest:** Ensures that every employee e has at least one free weekend every 4 weeks: constraints C4 and C5 in Figure 5.
- (9) **Night series A:** For every employee e a 48-hour rest period must follow after a series of duties containing at least 3 night duties: constraint C7 in Figure 5.
- (10) **Night series B:** Prohibits a day duty to be part of a series of duties that contains at least 3 night duties. This prevents employees from having to shift their sleep cycle too often.
- (11) **Maximum night duties:** Every employee e is assigned to no more than the maximum number of night duties every 4 weeks: constraint W10 in Figure 4.
- (12) **Lower bound worktime:** For every employee e , worktime less than their contract hours is counted as undertime.
- (13) **Upper bound worktime:** For every employee e , worktime exceeding the full-time equivalent is counted as overtime.
- (14) **Maximum overtime:** Every employee e cannot exceed the maximum amount of overtime per 4 weeks: constraint C2 in Figure 5.
- (15) **Maximum weekly hours:** Every employee e cannot exceed the maximum number of workhours per week: constraint W1 in Figure 4.
- (16) **Maximum number of duties:** Every employee e cannot exceed the maximum number of assignments per 4 weeks: constraint C1 in Figure 5.
- (17) $Y_{e,a,k}$ is a binary variable.
- (18) OT_e , UT_e , TH_e , and $WH_{e,w}$ are non-negative variables.

6.2. Column generation scheme

A column generation scheme consists of three big steps. First, a starting scenario has to be obtained. As a second step the scheme repeatedly improves the solution by adding new columns, or in this case individual schedules. Last of all, for every employee a schedule has to be selected from the pool of generated schedules.

Master model

Sets

- E** Employees, indexed by e
- A** Arcs, indexed by a
- A^{Duty}** Arcs related to duties, subset of **A**
- R** Individual schedules, indexed by r

Variables

- $X_{e,r}$ 1 If employee $e \in E$ carries out schedule $r \in R$
0 Otherwise
- U_a 1 If duty $a \in A^{Duty}$ is not covered by an assigned schedule $r \in R$
0 Otherwise

Parameters

- $a_{r,a}$ 1 If duty $a \in A^{Duty}$ is covered by schedule $r \in R$
0 Otherwise
- $b_{e,r}$ 1 If schedule $r \in R$ is associated with employee $e \in E$
0 Otherwise
- c_r Cost of schedule $r \in R$
- M** Large number

Minimize

$$\sum_{(e,r)} [X(e,r) * c(r)] + M * \sum_{a \in A^{Duty}} [U_a]$$

Subject to

- (1) $\sum_r [X_{e,r} * a_{r,a} * b_{e,r}] + U_a \geq 1 \quad \forall a \in A^{Duty}$
 - (2) $\sum_r [X_{e,r} * b_{e,r}] = 1 \quad \forall e \in E$
 - (3) $X_{e,r} \geq 0 \quad \forall e \in E, r \in R$
 - (4) $U_a \geq 0 \quad \forall a \in A^{Duty}$
- (1) **Coverage:** All duties $a \in A^{Duty}$ have to be covered by at least one assigned schedule or U_a has value 1.
 - (2) **Schedules:** All employees e are assigned to exactly one schedule in total.
 - (3) $X_{e,r}$ is a non-negative variable.
 - (4) U_a is a non-negative variable.

Figure 9, Master model

The master model is a simplified version of the entire model, because it only has information regarding schedules and not regarding individual duties and is therefore less complex. Its task is to select schedules for employees such that total costs are as low as possible. For the first run the Master model uses schedules obtained from the initial solution, which is straight forward, because there is just one schedule available per employee. After the first solve, new schedules are added to the schedule pool by the column generation scheme and the Master model has to choose which schedules to select for every employee. However, the Master model solves a relaxed version of the problem, usually resulting in a fractional solution. It is called a relaxed version, because constraint (3) in *Figure 9* does not require variable $X_{e,r}$ to have a value of either 0 or 1, but is allowed to be nonnegative. A fractional solution means that the model assigns parts or fractions of schedules to employees instead of assigning exactly one schedule to every employee. This allows the model to find a solution much quicker and gives a good lower bound for the cost of the solution, but it does not result in a feasible solution. Therefore, a procedure called Integer solution is used later in the process to obtain a feasible solution, where every employee is assigned to exactly one schedule.

Sub model

The core of the column generation scheme is still the network flow model as described in chapter 6.1. However, the model is adjusted to consider only one employee at a time to reduce complexity of the model, these adjustments are described in *Figure 10*. Whereas the network flow model was too complex to solve for the entire set of employees in a reasonable amount of time, the Sub model is solved very quickly: around 0.05 seconds per solve. With every solve, the Sub model generates an individual schedule that is in accordance with legislation, labor agreements and preferred scheduling rules.

Solve the network flow model as described in chapter 6.1 with several adjustments:

- Constraint (1): Coverage is not used, because not all duties need to be covered by a single employee. Coverage of duties is taken care of in the Master model.
- The network flow model is not solved for the set of all employees, but only for one employee at a time. This employee is referred to as CurrentEmployee and is defined in the Column generation procedure in *Figure 12*.
- The objective of the model is now to minimize reduced costs where α_a and β_e are dual multipliers obtained from the Master model (*Figure 9*) constraints (1) and (2) respectively.

Minimize:

$$Reduced\ costs = C_e - \sum_{k,a \in A^{Duty}} [Y_{e,a,k} * \alpha_a * S_{e,a}] + \beta_e$$

Figure 10, Sub model

Since the Sub model only considers one employee at a time, it does not have information regarding what duties are already assigned and consequently what duties should be in an employee's schedule. This is solved by obtaining dual multipliers from the Master model that is described in *Figure 9*. The Master model does consider all duties and employees and dual multipliers obtained from the Master model help to identify what duties are valuable to add in a new individual schedule. Simply said, the dual multiplier shows the marginal value of having an additional unit of a certain object in the Master model. To explain, a duty that is unassigned in the Master model results in a high penalty cost and will thus be valuable to add in a new individual schedule.

Using the dual multipliers obtained from the Master model, the Sub model now does have information regarding the relative value of every duty. Consequently, the Sub model can calculate what the relative value, or reduced costs, of a new individual schedule would be based on what duties are part of this new schedule. For every employee the Sub model generates a new individual schedule, optimized with regards to reduced costs. Since a duty needs to be assigned to only one employee, it is not helpful to add a lot of new schedules containing a certain duty. Simply adding all new schedules to the schedule pool would make the pool grow large very quickly. Therefore, only the n best individual schedules, with the most negative reduced costs, are selected to add to the schedule pool.

Main execution

```

Generate_initial_solution;      // Generates a schedule for every employee.
while LoopNr < n do
    Solve_Column_Generation;    // Generates new individual schedules and solves the
                                // fractional problem.
    Solve_Integer_Problem;      // Solves the Master problem, but now with  $X_{e,r} = \{0,1\}$ .
                                // With a time limit of 500 seconds.
    Fix_Employee_Schedules;     // Selects the 15 most efficient individual schedules
                                // and fixes them.
    LoopNr += 1;
endwhile;                      // When  $n \cdot 15$  employees are fixed, the problem is
                                // small enough to solve using an exact approach.
Fill_duties;                   // Assigns remaining duties.

If Problem is small enough then
    Fill_Duties for other part; // Shuffles duties of suitable employees to assign
Endif;                         // remaining duties

If unable to assign all duties then
    Run Main execution procedure again;
Endif;

```

Figure 11, Main execution procedure

The column generation scheme works on the principle of repeatedly improving the solution. In order to achieve this, several steps need to be taken and repeated. Therefore, a procedure called Main execution is used to call several procedures and to make sure these are executed in the correct sequence. The Main execution procedure is given in *Figure 11*. After the main execution procedure is finished, it is not guaranteed that an optimal solution has been found and improvement may be possible. Even though it is not the objective to find an optimal solution, the solution should be satisfactory. A solution that leaves duties unassigned is not satisfactory and therefore the Main execution procedure is run again if duties remain unassigned.

Solve column generation

The loop of the Main execution procedure (*Figure 11*) starts with a procedure called Solve column generation. The pseudo-code for this procedure can be found in *Figure 12*. This column generation procedure starts by solving the Master model found in *Figure 9* that assigns schedules to all employees. From this solution so called dual multipliers are obtained, α_a and β_e , from Master model constraints (1) and (2) respectively. These multipliers hold information about opportunities to improve the objective function. Using the dual multipliers, the sub model generates new schedules for every employee that are expected to improve the solution of the master model. The aim is to minimize the value of *Reduced costs* of every employee's schedule.

```

while 1.0 do "Solve-loop"
    solve Master model;
    Breaks "Solve-loop" if iterations > 20 and improvement Totalcost <= 1;
    Retrieve Duals;
    for (e in EmployeesToDo) do
        CurrentEmployee := e;
        Solve Sub model;
    endfor;
    Add n best individual schedules to column pool;
    Breaks "Solve-loop" if minimum reduced cost >= -0.5;
    Breaks "Solve-loop" after 150 iterations;
    Delete obsolete individual schedules;
    IterationIndex += 1;
endwhile;

```

Figure 12, Solve column generation procedure

However, it may appear that some individual schedules could not be improved, so no new columns are added to the schedule pool for these employees. Additionally, some schedules will have greater negative reduced costs than others and to improve efficiency off the model, only the n best schedules are added to the pool. To further improve efficiency, schedules that have not been used in the solution of the Master model in the last m iterations are considered

obsolete, meaning that better alternatives exist. These obsolete schedules are removed from the schedule pool with the aim to limit the size of the schedule pool and to keep the model from growing too large.

A loop lets the master and sub model be solved in sequence until a certain condition is reached. The loop stops when no improvement can be found, after a fixed number of iterations or if the reduction in *Totalcost* in the Master model is too small. When the loop stops, improvements are small and running the model longer yields minor benefit or a near optimal solution for the master model is found. This solution however, is often a fractional solution to the relaxed problem and further steps are required to obtain a usable integer solution. These steps are explained in *Figure 14*.

Initial solution

Even though it is not necessary to have a good starting point in order to obtain a good end solution, it does speed up the process of arriving at a good end solution, because the model does not have to make that many improvements. On the other hand, it is not effective to spend too much time on the initial solution. Therefore, a simple heuristic, able to assign a large part of the duties, is applied. Even though the aim is to assign as many duties as possible using this heuristic, night duties are left out the initial solution, because night duties have been observed to bring complex restrictions with regards to rest times.

```
for (e in ESortedSkills) do
  Add new individual schedule;
  Add unassigned duties to new individual schedule such that:
    - e Has the skill for these duties
    - Duties do not coincide with each other
    - Schedule has a weekend after every 5 duties
    - At most MaxD duties are in the schedule;
  Calculate ScheduleCost;
endfor;
```

Figure 13, Generate initial solution procedure

A simplified version, also called pseudo-code, of the procedure Generate initial solution can be found in *Figure 13*. The first step in generating an initial solution is to sort employees by the number of duties they are suited for. Then, starting with the employee that can be assigned to the least number of duties, duties are assigned to every employee. Similar to sorting employees, duties are sorted by the number of employees able to carry out this duty. When the procedure has passed all employees, the loop stops and the next procedure in the Main execution is called. Information regarding what duties every employee is assigned to is stored as schedules. To explain, the Generate initial solution procedure creates a schedule for every employee and adds duties to that schedule following the procedure described in *Figure 13*.

Integer solution

After the column generation is finished, the next step is to obtain a feasible integer solution (Figure 14) that can be applied in practice, because a solution obtained by the column generation procedure is often a fractional solution. In an integer solution every employee is assigned to exactly one schedule. This is achieved by adjusting constraints (3) and (4) of the Master model. Constraints (3) and (4) now restrict the decision variables to be binary, which increases the complexity of the problem. To keep the complexity of the integer model limited, obsolete schedules are removed from the schedule pool as described in Solve column generation. Even though the size of the schedules pool is limited, the integer solution proves difficult to solve in early iterations of the Column generation procedure. Therefore, a time limit of 500 seconds is set for the integer model. If the algorithm reaches 500 seconds and has not found the optimal solution, the best feasible solution found is used. This approach is chosen because the gap of the integer solution was in some occasions still over 90% after 3000 seconds. Instead of waiting for optimal integer solutions during every iteration and achieving better solutions, the improving capabilities of the scheme are used to correct for potentially bad intermediate solutions and decisions.

Solves the Master model as described in Figure 9 with one adjustment:

- Constraint (3) and (4): Decision variables $X_{e,r}$ and U_a are no longer nonnegative variables, but are now binary variables. This means that constraints (3) and (4) in the Master model now read:

$$(3) \quad X_{e,r} = \{0,1\} \quad \forall e \in \mathbf{E}, r \in \mathbf{R}$$

$$(4) \quad U_a = \{0,1\} \quad \forall a \in \mathbf{A}^{Duty}$$

Figure 14, Integer solution procedure

Fix employee schedules

A schedule is selected for every employee by the integer problem. These schedules are scored based on their efficiency. This efficiency is determined based on the duties covered by that schedule. A duty that can be carried out by a small number of remaining employees has a higher value than a duty that has a lot of suitable employees left. Additionally, duties that have a lot of overlap with other duties also have high value. If two duties have overlap it means that they cannot be carried out by the same person due to coinciding work times. Thus, an efficient individual schedule covers duties with high value. As said all schedules assigned to employees are scored on their efficiency, the fifteen best scoring schedules are now fixed by the Fix employee schedules procedure described in Figure 15. Employees assigned to these fixed schedules, as well as the duties covered by these schedules, are no longer considered in the main execution loop.

```

for (e in EmployeesSortedEfficiency) do
    if nr<=15 then
        Fix employee e;
        Fix duties assigned to employee e;
        nr += 1;
    endif;
endfor;

```

Figure 15, Fix employee schedules procedure

Fill duties

After the column generation procedure has run 8 times and 120 ($8 \times 15 = 120$) employees are fixed, 25 employees and around 125 duties remain to be assigned. This is small enough to solve using an exact approach. Therefore, the network flow model as described in chapter 6.1 that assigns duties to employees is solved with adjustments found in Figure 16. This model takes into account all labor agreements and legislation and assigns remaining duties to available employees while minimizing total costs. An available employee refers to either an employee that is not fixed yet, or a fixed employee with a gap in their schedule that can be used to assign remaining duties. Considering all employees results in a large problem, but not too large to solve within an hour, because the biggest part of the schedules is already determined by the previously assigned duties. Moreover, considering all employees in this fashion helps the model to reduce the number of unassigned duties.

It may also occur that some of the fixed duties are assigned to multiple employees, while only one employee is required. The fact that duties can be assigned to multiple employees is a result of the integer solution, because the integer problem is solved with a relaxed coverage constraint, allowing duties to be assigned more than once. In that case the model selects only one employee to carry out the duty, leaving a gap in the other employees' schedule which can be used to carry out a different duty. To summarize, the procedure Fill duties starts by assigning duties that were fixed in an earlier step. Some of these fixed duties may be assigned to multiple employees. In this case, the procedure selects only one of those employees. Duties that were not fixed are assigned to employees while taking into account legislation and other scheduling rules.

Fixing the majority of duties greatly reduces the complexity of the problem and allows the network flow model to be solved to optimality within an hour. However, fixing duties can be too restrictive to find a solution where all duties are assigned. Therefore, a list is made of the unassigned duties and employees suitable for these duties are selected. If there are more than 10 unassigned duties or more than 20 suitable employees in this list, the found solution is used as a new initial solution and the column generation scheme is executed again. If the solution shows 10 or less unassigned duties and 20 or less suitable employees a different part of the model is fixed and the procedure Fill duties is executed again. The newly fixed part consists of employees not suitable to carry out the unassigned duties, the duties assigned to these now

fixed employees are fixed as well. This means that the Fill duties procedure now only has to consider employees suitable for the unassigned duties in terms of employees. In terms of duties, the procedure considers the unassigned duties as well as duties that were assigned to the suitable employees. In fact, the model shuffles the duties of these selected employees and attempts to fit in the unassigned duties. This method is able to find a solution where no unassigned duties are left.

Solve the network flow model as described in chapter 6.1 with one adjustment:

- An additional constraint is considered. This constraint ensures that exactly one employee is assigned to all fixed duties. The model has to choose from one or multiple employees assigned to those duties as done earlier by the integer solution. The additional constraint reads:

$$\sum_{k,e \in E^{Fixed}} [Y_{e,a,k} * S_{e,a} * ED_{e,a}] = 1 \quad \forall a \in A^{DutyFixed}$$

Where parameter $ED_{e,a}$ equals 1 if duty a from the set of fixed duties is assigned to employee e and 0 otherwise.

Figure 16, Fill duties procedure

6.3. Model extensions

The models in chapter 6.1 and chapter 6.2 describe the situation of EBN in a sufficient matter. However, these models could be extended when management desires to include more information in their decision making or to model a situation at a different company. Several extensions will be described and explained in this chapter.

Travel costs

An addition to the network flow model could be travel costs for personnel. This refers to the compensation employees receive from their employer for traveling to and from work. The compensation is based on distance travelled and consequently costs increase if employees are assigned to duties further from their home. If one wants to consider traveling costs, a parameter containing information regarding travel costs for every employee to all different duties can be added to the network flow model presented in chapter 6.1. Thereafter, a new term can be introduced in the objective function that adds travel costs for the assigned duties. Mathematical formulations of this extension are presented in *Figure 17*.

A new parameter has to be introduced:

$trc_{e,a}$ Travel costs related to employee $e \in E$ carrying out duty $a \in A^{Duty}$

An additional term is added to the objective function:

$$+ \sum_{e,k,a \in A^{Duty}} [trc_{e,a} * Y_{e,a,k} * q_5]$$

This term sums over all employees, preference penalty indicators and arcs related to duties. Travel costs are now summed if employee $e \in E$ carries out duty $a \in A^{Duty}$ and multiplied by parameter q_5 to indicate the weight of these costs.

Figure 17, Travel costs

Preferred days off

Employees may have certain days on which they prefer to have a day off. In order to include this in the model, new arcs have to be added that refer to a single day off. This requires one arc for each day, thus 28 new arcs. These day off arcs are a subset of the entire set of arcs. Additionally, a new parameter is required that holds information for every employee on which days they prefer to be free. Agreements should be made on what rules apply for employees stating their preferred days off. Depending on the situation, preferred days off can be modelled as hard or soft constraints. If modelled as a hard constraint, the solution must always be in accordance with preferred days off. If modelled as a soft constraint, preferred days off can be disregarded at the cost of a penalty in the objective function. The extension of preferred days off is presented in *Figure 18*.

New arcs have to be introduced. One arc for each day, starting at 0:00h and ending at 24:00h. These 28 arcs are part of the new set \mathbf{A}^{DayOff} subset of \mathbf{A} .

A new parameter has to be introduced:

$$\begin{aligned} pdo_{e,a} &= 1 \text{ If employee } e \in \mathbf{E} \text{ prefers to have a day off on day } a \in \mathbf{A}^{DayOff} \\ &= 0 \text{ Otherwise} \end{aligned}$$

Option 1: Hard constraint

If preferred days off have to be modelled as a hard constraint, a new constraint has to be introduced:

$$\sum_k [Y_{e,a,k}] \geq pdo_{e,a} \quad \forall e \in \mathbf{E}, a \in \mathbf{A}^{DayOff}$$

This constraint ensures that every employee is assigned to a day off if they have marked that day as a preferred day off.

Option 2: Soft constraint

If preferred days off have to be modelled as a soft constraint, an additional term is added to the objective function:

$$+ \sum_{e,k,a \in \mathbf{A}^{DayOff}} [(1 - Y_{e,a,k}) * pdo_{e,a} * q_6]$$

This term sums over all employees, preference penalty indicators and arcs related to days off. A penalty of q_6 is now added to the objective function if employee $e \in \mathbf{E}$ prefers a day off on day $a \in \mathbf{A}^{DayOff}$ and is not assigned to this day off. To explain, if employee e is not assigned to day off a , $Y_{e,a,k}$ has value 0 and $(1 - Y_{e,a,k}) = 1$. Since $pdo_{e,a}$ has value 1 as well, the equation becomes $(1 - 0) * 1 * q_6 = 1 * 1 * q_6 = q_6$. If $pdo_{e,a}$ equals 0 or both $Y_{e,a,k}$ and $pdo_{e,a}$ are equal to 1, no penalty is added to the objective function.

In case it is desired to have different penalties for different employees and/or days off, parameter $pdo_{e,a}$ can be adjusted to display these differences:

$$\begin{aligned} pdo_{e,a} &\geq 1 \text{ If employee } e \in \mathbf{E} \text{ prefers to have a day off on day } a \in \mathbf{A}^{DayOff} \\ &= 0 \text{ Otherwise} \end{aligned}$$

Note that $pdo_{e,a}$ is now greater than 1 to indicate preferred days off. A higher value means more weight is given to disregarding a certain preferred day off.

Figure 18, Preferred days off

Preferred duties

In a similar fashion to preferred days off, employees may have duties they prefer to be assigned to or duties they prefer not to be assigned to. Again, a new parameter is required that holds information regarding these preferred and/or undesired duties. Adjustments to the network flow model in chapter 6.1 are presented in *Figure 19* for preferred duties and *Figure 20* shows adjustments for undesired duties.

A new parameter has to be introduced:

$$\begin{aligned} pd_{e,a} &= 1 \text{ If employee } e \in \mathbf{E} \text{ prefers to be assigned to duty } a \in \mathbf{A}^{Duty} \\ &= 0 \text{ Otherwise} \end{aligned}$$

Option 1: Hard constraint

If the preferred duties rule has to be modelled as a hard constraint, a new constraint has to be introduced:

$$\sum_k [Y_{e,a,k}] \geq pd_{e,a} \quad \forall e \in \mathbf{E}, a \in \mathbf{A}^{Duty}$$

This constraint ensures that every employee is assigned to a duty if they have marked that duty as preferred.

Option 2: Soft constraint

If the preferred duties rule has to be modelled as a soft constraint, an additional term is added to the objective function:

$$+ \sum_{e,k,a \in \mathbf{A}^{Duty}} [(1 - Y_{e,a,k}) * pd_{e,a} * q_7]$$

This term sums over all employees, preference penalty indicators and arcs related to duties. A penalty of q_7 is now added to the objective function if employee $e \in \mathbf{E}$ prefers duty $a \in \mathbf{A}^{Duty}$ and is not assigned to this duty. To explain, if employee e is not assigned to duty a , $Y_{e,a,k}$ has value 0 and $(1 - Y_{e,a,k}) = 1$. Since $pd_{e,a}$ has value 1 as well, the equation becomes $(1 - 0) * 1 * q_7 = 1 * 1 * q_7 = q_7$. If $pd_{e,a}$ equals 0 or both $Y_{e,a,k}$ and $pd_{e,a}$ are equal to 1, no penalty is added to the objective function.

In case it is desired to have different penalties for different employees and/or duties, parameter $pd_{e,a}$ can be adjusted to display these differences:

$$\begin{aligned} pd_{e,a} &\geq 1 \text{ If employee } e \in \mathbf{E} \text{ prefers to be assigned to duty } a \in \mathbf{A}^{Duty} \\ &= 0 \text{ Otherwise} \end{aligned}$$

Note that $pd_{e,a}$ is now greater than 1 to indicate preferred duties. A higher value means more weight is given to disregarding a certain preferred duty.

Figure 19, Preferred duties

A new parameter has to be introduced:

$$\begin{aligned} ud_{e,a} &= 1 \text{ If employee } e \in \mathbf{E} \text{ prefers not to be assigned to duty } a \in \mathbf{A}^{Duty} \\ &= 0 \text{ Otherwise} \end{aligned}$$

Option 1: Hard constraint

If the undesired duties rule has to be modelled as a hard constraint, a new constraint has to be introduced:

$$\sum_k [Y_{e,a,k} * ud_{e,a}] < 1 \quad \forall e \in \mathbf{E}, a \in \mathbf{A}^{Duty}$$

This constraint ensures that an employee is not assigned to a duty if they have marked that duty as undesired. To further explain, in this equation one of the terms is allowed to have a value of 1, because if both terms are equal to 1 the left-hand side of the equation is no longer smaller than 1.

Option 2: Soft constraint

If the undesired duties rule has to be modelled as a soft constraint, an additional term is added to the objective function:

$$+ \sum_{e,k,a \in \mathbf{A}^{Duty}} [Y_{e,a,k} * ud_{e,a} * q_8]$$

This term sums over all employees, preference penalty indicators and arcs related to duties. A penalty of q_8 is now added to the objective function if employee $e \in \mathbf{E}$ prefers not to be assigned to duty $a \in \mathbf{A}^{Duty}$ and is to this duty. To explain, if one of $Y_{e,a,k}$ or $ud_{e,a}$ is 0, an employee is not assigned to an undesired duty and no penalty is added. However, if both $Y_{e,a,k}$ and $ud_{e,a}$ are equal to 1 an employee is assigned to an undesired duty and a penalty of q_8 is added to the objective function.

In case it is desired to have different penalties for different employees and/or duties, parameter $ud_{e,a}$ can be adjusted to display these differences:

$$\begin{aligned} ud_{e,a} &\geq 1 \text{ If employee } e \in \mathbf{E} \text{ prefers not to be assigned to duty } a \in \mathbf{A}^{Duty} \\ &= 0 \text{ Otherwise} \end{aligned}$$

Note that $ud_{e,a}$ is now greater than 1 to indicate undesired duties. A higher value means more weight is given to disregarding a certain undesired duty.

Figure 20, Undesired duties

7. Analysis

This chapter will discuss the results obtained from the model presented in chapter 6.2. All calculations have been performed on a desktop PC with an Intel core i5 quad core processor running at 3,3GHz-3,7GHz with 8GB of RAM memory. The CPLEX 12.8 solver in AIMMS is used to perform all calculations. A table containing a summary of the results is presented in *Figure 21*. The contents of this table will be explained and discussed throughout this chapter.

Runs

The first column of *Figure 21* refers to the number of times the model has been executed. Run A1 to A8 each represent one loop of the Main execution procedure presented in *Figure 11*. Each loop consists of running the column generation scheme, solving the integer problem and fixing 15 employees. After run A8, the Fill duties procedure presented in *Figure 16* was executed of which the results are found at run AFill. After run AFill too many suitable employees were left for the unassigned duties meaning that the problem was too large to solve for unassigned duties within reasonable time. Therefore, the solution found in run AFill was used as the initial solution for a second execution of the Main execution procedure. The loops of the second execution are called B1 to B8 and after this loop, problems BFill, FillOP and FillOP2 were solved. Runs B1 to B8 are similar to runs A1 to A8, with the difference that run B1 uses the solution of AFill as initial solution, whereas A1 used an initial solution generated by the heuristic presented in *Figure 13*. In run BFill the Fill duties procedure was executed again, but now with results from run B8. After BFill, one duty remained unassigned and run FillOP was executed with the aim to assign the remaining duties. FillOP solved the same model as BFill, but all employees were fixed, except employees suited to carry out the unassigned duty. Again, one duty remained unassigned and FillOP2 was executed with the same logic as FillOP. This resulted in a solution where no duties were left unassigned and no preference scheduling rules were violated. However, there remained overtime and undertime in the solution, meaning that there may remain room for improvement.

Solving time

Looking at *Figure 21*, the first thing that should be noted is the solving time of each run. The second column of the table shows the total time of the loop, which consists of the solving time of the column generation procedure found in the fifth column and the integer problem solution time found in the tenth column. In general, the solving times decrease with each loop, which is explained by the fact that the problem size decreases with every loop, because part of the active employees and duties are fixed after every loop. The third and fourth column show the number of active employees and active duties respectively, referring to employees and duties that are not fixed and are thus considered by the models. As can be seen, the algorithm starts with 145 active employees and 2040 active duties and every loop 15 employees and their assigned duties are fixed based on their efficiency.

1				Column generation					Integer solution				
	2	3	4	5	6	7	8	9	10	11	12	13	14
Run	Total time (s)	Active employees	Active duties	Time (s)	Iterations	Columns added	Solution first iteration	Solution final iteration	Time (s)	Columns used	Solution	Gap	#Duties not assigned
A1	1200	145	2040	700	22	2930	5.2100E+12	4.2857E+09	500	945	1.2400E+12	90.72%	124
A2	994	130	1740	494	23	2380	2.9427E+10	2.1019E+04	500	961	1.5400E+12	90.51%	154
A3	1098	115	1460	598	25	1769	6.6182E+10	4.0000E+10	500	939	1.4800E+12	84.29%	148
A4	1022	100	1191	522	22	1458	1.0405E+11	4.0000E+10	500	852	1.3600E+12	82.72%	136
A5	896	85	929	396	22	1342	4.5429E+11	6.0000E+10	500	779	9.7000E+11	79.51%	97
A6	875	70	675	375	28	1473	4.9295E+11	7.0000E+10	500	657	5.6000E+11	74.52%	56
A7	436	55	457	227	36	1053	3.0106E+11	1.1000E+11	209	377	2.7000E+11	0%	27
A8	70	40	264	70	25	667	2.6135E+11	1.3000E+11	0.09	200	1.4000E+11	0%	14
AFill	3000	25	123	-	-	-	-	-	3000	-	5.0000E+10	<0.01%	5
B1	486	145	2040	473	23	2919	5.0000E+10	1.9974E+04	13	847	1.0000E+11	0%	10
B2	368	130	1754	358	25	2164	1.7709E+04	1.6995E+04	9.51	802	1.0000E+11	0%	10
B3	406	115	1465	352	28	1333	1.4854E+04	1.4573E+04	54	632	2.4000E+11	0%	24
B4	503	100	1197	416	35	1485	1.2606E+10	1.2294E+04	87	630	2.7000E+11	0%	27
B5	356	85	922	317	36	1485	9.1721E+10	1.0402E+04	39	487	2.5000E+11	0%	25
B6	296	70	676	211	26	1067	8.6918E+10	8.2592E+03	85	419	2.1000E+11	0%	21
B7	148	55	435	148	32	1284	2.1000E+11	2.0000E+10	0.3	316	5.0000E+10	0%	5
B8	57	40	260	57	23	648	5.0000E+10	3.0000E+10	0.09	964	3.0000E+10	0%	3
BFill	636	25	114	-	-	-	-	-	636	-	1.0000E+10	0%	1
FillOP	62	3	39	-	-	-	-	-	62	-	1.0000E+10	0%	1
FillOP2	147	7	128	-	-	-	-	-	147	-	556	0%	0
EBN	-	-	-	-	-	-	-	-	-	-	2.5000E+11	-	0

Figure 21, Results

For the integer problem, a time limit of 500 seconds was set, resulting in a solution gap for runs where the integer problem reached 500 seconds. Gap refers to the percentage difference between the best found solution and a lower bound calculated by the CPLEX 12.8 solver. In order to reduce solving times and gaps, the algorithm is set to limit the number of columns and delete obsolete columns as described in chapter 6.2. Column number 7 shows the number of columns generated by the Column generation procedure during that run and column number 11 shows the number of columns remaining after obsolete columns are deleted. This pool of remaining columns is the column pool that is used by the integer problem to find an integer solution.

Solving time is not only affected by the problem size, the complexity of the problem plays a part as well. This is illustrated by comparing the solving time of run A1 and run B1, two problems of similar size. A major difference though is that the initial solution of A1 was constructed by a simple heuristic where night duties were not considered, resulting in a poor solution and evidently a complex problem. In contrast, the initial solution of run B1 originates from the solution of AFill which is of much better quality. To compare, A1 started with more than 500 unassigned duties, whereas B1 started with less than 10 unassigned duties. The complexity plays a big part, since the integer solution of A1 took 500 seconds to reach a gap of around 90%, while the integer problem of B1 found an optimal solution in 13 seconds.

Solutions

The last part of *Figure 21* that requires an explanation is the columns containing solution values. Column eight and nine present the first solution and the last solution of the column generation scheme. This is to illustrate in short how much improvement was found by the column generation. An extended table containing solutions found in every iteration is presented in *Figure 25* in Appendix 11.2 and *Figure 26* in Appendix 11.3. Column 12 of the table in *Figure 21* contains the solutions found by the integer problem using the column pool generated by the column generation scheme. The values found for solutions in *Figure 21* are calculated by summing up several factors. First of all, the overtime and undertime of all employees are summed and multiplied by penalty coefficient Q_1 and Q_2 respectively. Next, the total number of unassigned duties, found in column 14, is multiplied by a large penalty coefficient Q_3 . Last of all, the number of violations regarding preferred scheduling rules are multiplied by a moderately sized penalty coefficient Q_4 . These penalty coefficients are set to values based on interviews with management where the importance of every factor was determined. It was established that unassigned duties are most important to prevent, followed by violations of preferred scheduling rules and ultimately overtime and undertime are equally important. Mathematically stated, the following equation holds for penalty coefficients Q_1 to Q_4 :

$$Q_3 > Q_4 > Q_1 = Q_2$$

The table in *Figure 21* shows that the solution after the final iteration of the column generation scheme is always better than the solution after the first iteration. This is no coincidence, because columns with negative reduced costs are repeatedly added to the column pool and only obsolete columns are removed. Therefore, the algorithm should often be able to find a solution that is at least as good as the previous solution and should in the long run be able to find improvements. If this is not the case, there may be a problem with the algorithm or an optimal solution is already found and improvement is not possible. Furthermore, the solution to the integer problem can only be as good as or worse than the solution found by the column generation scheme, because the integer problem solves a more restricted problem using the same columns.

Even though iterations in a single column generation loop should show improvements, this is not the case when comparing subsequent runs of the column generation scheme. Column 9 in *Figure 21* shows that solutions after the final iteration increase for run A2 to A8, meaning that the quality of the solutions worsens with every run. This is caused by the fact that after every run, part of the employees and duties are fixed and are no longer considered in the model. This could mean that decisions made by the model have negative effects on the results of later runs. For example, a duty may no longer be assignable because all suitable employees are already fixed and cannot be assigned to new duties. Even though this is undesired behavior of the model and could possibly be prevented by allowing longer solving times, these results are taken for granted and the problems were solved in a later stage by filling duties and by

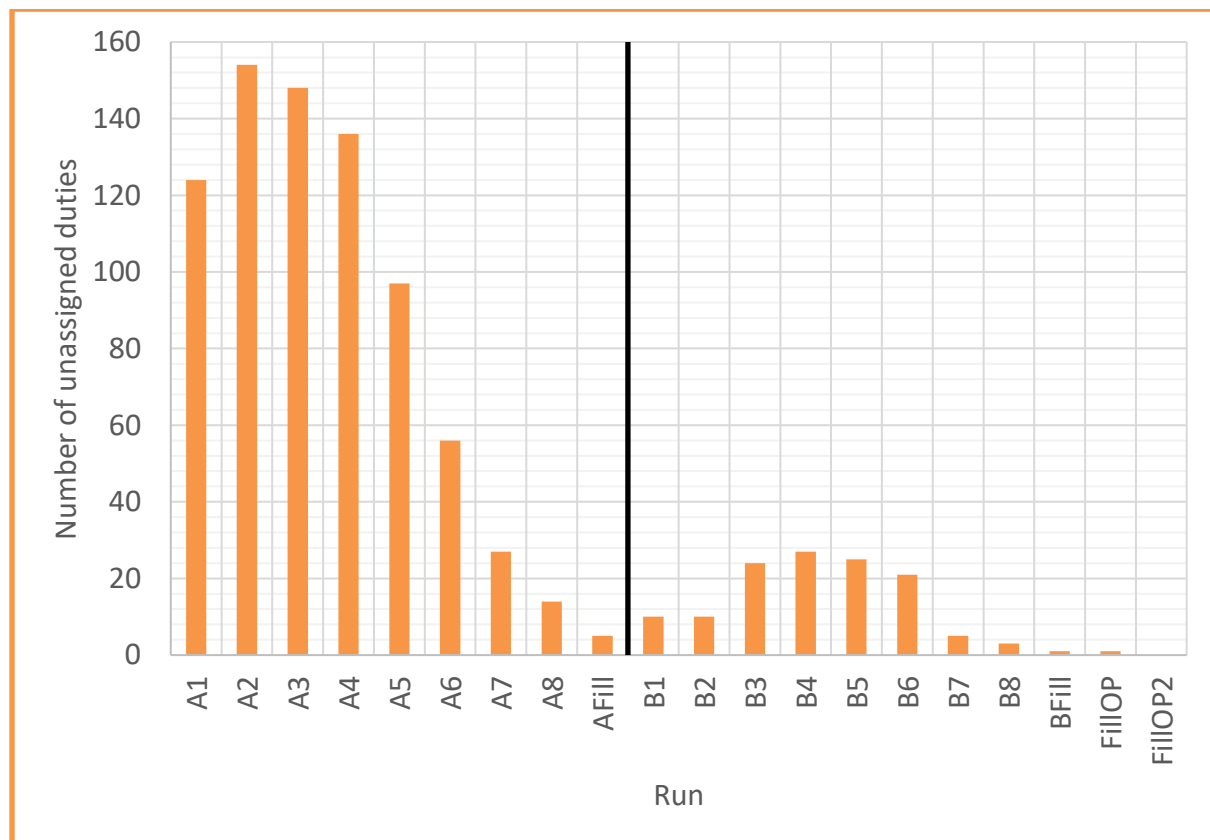


Figure 22, Number of unassigned duties

running the entire model a second time. The same is observed by looking at the integer solutions presented in column 12 and the biggest cost factor: the number of unassigned duties which can be found in column 14. A graphical representation of the number of unassigned duties in the integer solution of every run is found in *Figure 22*. The black vertical line at the center of the graph indicates the end of the first run of the model. It is observed that in both runs the number of unassigned duties increases at first but steadily decreases after both peaks.

Furthermore, it should be noted that the solution of the first iteration of run BI is equal to the solution of AFill. It is no surprise that these values are exactly the same, because the solutions consist of exactly the same individual schedules. This is on purpose, because the solution of AFill is used as initial solution for BI with the aim to improve the solution of AFill. This approach has succeeded in improving the solution of AFill, proven by the fact that the solution of BFill is 5 times smaller than the solution of AFill. Additionally, the size of the model has reached a level that allows solving the remaining duties in run FillOP. Even though the solution of FillOP did not immediately show a large improvement compared to BFill, there was now a different duty left unassigned, allowing to solve a different part of the model in run FillOP2. This resulted in a solution where no duties were left unassigned and no preference scheduling rules were broken. An example of an individual schedule generated by the model can be found in *Figure 23*. Light blue blocks represent day duties, dark blue blocks represent night duties, white blocks represent weekly 36-hour rests and grey blocks represent two days off. Since the schedule is designed to be cyclic, the employee assigned to this individual schedule can repeat this schedule without breaking any rules. To explain, after week 4 the employee starts again with week 1. *Figure 24* shows an example of a duty that has to be carried out every Monday to Friday. This duty may be carried out by a different employee every day as shown in the example.

To see how the results found in this research improve the aggregate schedule of EBN, the last two rows of *Figure 21* should be compared. The row called EBN shows the penalty costs of the aggregate base schedule that is currently used at EBN. Costs of the current schedule are calculated using the same penalty coefficients that were used to calculate costs for the newly generated schedules. Looking at column 14, it shows that both schedules leave 0 duties unassigned which is the most important cost factor. However, the current schedule of EBN does not conform to all preferred scheduling rules for every employee. This results in penalty costs equal to $2.5000E+11$ and is much higher than the costs of the newly generated aggregate schedule in run FillOP2 which are equal to 556. Run FillOP2 yielded a schedule that conforms to all preferred scheduling rules and leaves no duties unassigned. The remaining costs consist of undertime and no overtime. The generated schedule is a base schedule and undertime will (partly) further decrease in practice, because clients typically request additional services and employees may call in sick or take days off.

	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
	October 1	2	3	4	5	6	7
Week 1	Duty A1 16:00-24:00	Weekly rest 36H 0:00-12:00		Duty B3 18:00-01:00	Duty B4 18:00-01:00	Weekly rest 36H 01:00-13:00	Duty C6 14:00-23:00
	8	9	10	11	12	13	14
Week 2	Duty E6 18:00-01:00	Duty E7 18:00-01:00	Weekly rest 36H 01:00-13:00		Duty E9 18:00-01:00	Duty E10 17:00-01:00	Duty F13 22:00-07:00
	15	16	17	18	19	20	21
Week 3	Weekly rest 36H 07:00-19:00		Duty F16 22:00-08:00	Duty H17 23:00-08:00	Duty I18 23:00-08:00	Weekend 00:00-24:00	
	22	23	24	25	26	27	28
Week 4	Duty F22 22:00-08:00	Duty I23 23:00-08:00	Duty I24 23:00-08:00	Duty G19 22:00-07:00	Weekend 00:00-24:00		

Legend:

Day duty	Night duty	Weekly rest 36H	Weekend 00:00-24:00
----------	------------	--------------------	------------------------

Figure 23, Example of an individual schedule

	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
	October 1	2	3	4	5	6	7
Week 1	Duty A1 Empl. 1	Duty A2 Empl. 2	Duty A3 Empl. 2	Duty A4 Empl. 3	Duty A5 Empl. 4		
	8	9	10	11	12	13	14
Week 2	Duty A6 Empl. 5	Duty A7 Empl. 7	Duty A8 Empl. 8	Duty A9 Empl. 9	Duty A10 Empl. 10		
	15	16	17	18	19	20	21
Week 3	Duty A11 Empl. 11	Duty A12 Empl. 3	Duty A13 Empl. 12	Duty A14 Empl. 5	Duty A15 Empl. 12		
	22	23	24	25	26	27	28
Week 4	Duty A16 Empl. 3	Duty A17 Empl. 13	Duty A18 Empl. 13	Duty A19 Empl. 14	Duty A20 Empl. 4		


Legend:  Day duty

Figure 24, Example of duty schedule

8. Conclusion

In this chapter a conclusion will be drawn from this research project and the research questions will be answered. The research question and sub-questions will be repeated and then answered based on the results of this research.

Research question

How can the scheduling process at EBN be improved by developing and implementing a scheduling tool that applies an optimization algorithm?

Finding a feasible solution to the scheduling problem is a difficult task. There are many rules and agreements coming from legislation, labor agreements and preferences that have to be taken into account. Especially the fact that an adjustment in one employee's schedule can have extensive implications for schedules of other employees, makes the scheduling problem a complex task. For the planning department it is time consuming to construct a feasible aggregate schedule that covers all duties and is in accordance with legislation, let alone to optimize the schedule for preference rules and costs. To help the planning department, a column generation scheme was developed that is able to generate feasible individual schedules and is able to reduce costs including penalties for disregarding preferred scheduling rules. Result of this research project is a scheduling tool that can generate aggregate schedules without violating legislation and preference rules and is able to minimize overtime and undertime in the relaxed problem. This is an improvement to the current situation, because, as described in chapter 7, total penalty costs were reduced by a considerable amount. The actual costs of the solution consist of 0 hours of overtime and 556 hours of undertime. It was not a priority to find an optimal solution in terms of actual costs, because the end result is a base schedule. In practice, additional duties are requested and undertime will be solved.

Sub-questions

What are the requirements of the aggregate workforce schedule?

First of all, schedules must be in accordance with legislation as described in the Working Hours Act (Rijksoverheid, 2010) of which a summary can be found in appendix 11.1. Closely related to legislation are labor agreements defined in the CAO (De Nederlandse Veiligheidsbranche, 2017). These agreements have to be taken into account as well. Furthermore, all duties have to be covered by an employee that is suited for carrying out that duty. Last of all, total costs of the aggregate schedule should be minimized. These costs include salaries, penalties for violating preferred scheduling rules and penalties for leaving duties unassigned.

Is it necessary to use a cyclic schedule, if so what is the optimal cycle duration?

Strictly taken it is not necessary to use a cyclic schedule. However, if schedules are cyclic, it allows schedules to be structured and allows for schedules to be copied in subsequent periods. Moreover, a lot of rules as well as the salary payment are based on a 4-week period. Therefore, it is convenient to construct cyclic schedules.

What is the structure of duties? Does it allow flexibility in the work schedule, or does it force certain work sequences?

Figure 7 and Figure 8 show that some employees are only suited for a limited number of duties and that a few duties can only be carried out by a limited number of employees. This forces some individual schedules into a certain structure, but the majority of employees and duties allow for enough flexibility.

How to deal with time windows? Is it necessary to schedule time windows in such a fashion that the assigned duties never violate constraints?

The model developed in this research is not influenced by time window characteristics, because the option to swap duties among employees for ad-hoc planning is not considered in this research project. Therefore, time windows can be assigned freely to cover duties and do not need to be scheduled in such a manner that time windows on itself follow legislation and scheduling rules. Merely the assigned duties within the time windows are required to follow these rules. It may however be a valuable extension of the model to optimize flexibility in the ad-hoc scheduling phase as is further explained in chapter 9.

Which algorithm fits the situation of EBN?

An exact approach proved to be too complex due to the size and complexity of the problem. Therefore, a column generation scheme was applied, because of its applicability and previous successes in literature. This method showed satisfying results and fulfilled all objectives and thus fits the situation of EBN.

What are appropriate objectives for an optimizing algorithm? Does the objective function consist of costs, number of employees, job satisfaction, other factors or a mix of the previous factors?

The objective for optimization is to minimize the number of unassigned duties, violations of preferred scheduling rules and costs resulting from overtime and undertime.

Which requirements and objectives should be modelled as hard constraints, and which should be modelled as soft constraints?

Rules coming from legislation and labor agreements are first priority requirements and have to be modelled as hard constraints. Preferred scheduling rules are not a strict requirement and are allowed to be disregarded at the cost of a moderate penalty. Unassigned duties are undesirable but could result in infeasible solutions if this is modelled as a hard constraint. Therefore, unassigned duties are modelled as a soft constraint with very high penalty costs to ensure that as many duties as possible are assigned.

How can the tool be used to assist long term decision making on the tactical and/or strategic level?

The developed scheduling tool can be used to construct aggregate schedules based on different scenarios by adjusting input data and parameters without the need for major adjustments to the model. This allows management to compare schedules and their results in different scenarios and make decisions accordingly.

How does the developed tool improve the scheduling process, compared to the current scheduling process?

The developed scheduling tool allows the planning department to construct a base schedule within less time and with no penalty costs. Whereas it would take a planner multiple days to construct a base schedule, the algorithm is able to construct a base schedule in around 4 hours. Regarding the penalty costs associated with violating the preferred scheduling rules stated in chapter 2.2, the algorithm achieved great reductions as well. The current schedule of EBN contains 50 violations as shown in *Figure 6*. Considering that there are around 2000 duties in the base schedule, this means that there is a violation in 2.5% ($\frac{50}{2000} * 100 = 2.5$) of the duty assignments. The algorithm constructed a base schedule with no violations of the preferred scheduling rules.

9. Future research

Even though the research question and its related sub-questions have been answered, there are always issues that could be investigated further. This chapter will discuss topics that are interesting to look into if one desires to expand the research and reach beyond the scope of this project by investigating problems that are closely related to the research question.

Rotation of employees

An extension of the algorithm that deals with the rotation of duties for employees could be very helpful. The idea is that it could be beneficial to make sure that employees carry out certain duties at least a few times per 4-week period to keep their knowledge and experience required for that duty up to date. From the client's point of view, it could be beneficial as well to look into the rotation of guards, because several clients prefer a small selection of guards at their work site. The schedule generating model can be restricted to allow only a certain number of different employees to carry out duties related to clients with these preferences. Further research is required to find out which clients have such preferences and how this extension can be implemented in the model.

Hiring/firing

Next, it may be possible to extend the tool with a function that allows the user to define a ratio between the different employment types (full-time, part-time and temporary). Comparing the outcomes with different settings of this function, allows management to make better staff decisions. However, this could have major implications for the employee pool and the costs related to hiring and firing should be investigated thoroughly. A less radical option is to develop an extension that can show the implications of firing a certain employee or hiring a person with a certain set of skills.

Ad-hoc

A tool that helps the planning department to solve ad-hoc problems may be another helpful extension. This tool could, for example, show the planner what the best options are to solve a scheduling problem. A scheduling problem arises, for example, when an employee calls in sick or when a client requests an additional duty to be carried out. In this case the planning department has to figure out how this gap can be filled. An important issue to consider is that during the ad-hoc phase, employees have received their time window schedules and that additional costs may be related to making adjustments to the individual schedule. Moreover, it is not always possible to fill a gap by assigning an off-duty employee and multiple swaps may be necessary to solve the scheduling problem. A tool that can find solutions to these ad-hoc problems in a few minutes would help the planning department to quickly solve these issues in an efficient way. However, the problem remains that employees have to be asked to take on an additional or different duty and might refuse to do so.

Availability

Continuing on the ad-hoc issue discussed above, the availability of employees to fill gaps in the aggregate schedule may be optimized. As said, additional costs may be involved when assigning employees to duties that are not covered by their time window schedule. As a first step, time windows are assigned to cover at least all duties in the base schedule. However, some flexibility remains in assigning time windows, because duties may have a duration shorter than ten hours or an employee may have less than 20 duties in a 4-week period. This allows the planning department to schedule extra availability that can be used in case of ad-hoc problems. Further research is required to develop a method to optimize the availability of employees and to identify what the objective should be for optimization. Examples of optimization objectives are to have a back-up opportunity for as many duties as possible, or to have as many back-up opportunities as possible for certain high priority duties.

Improvement heuristic

Last of all, there may be opportunities to improve the solution generated by this research, because it is not proven that the solution is optimal. A good way to improve the solution may be to apply one or multiple improvement heuristics. These heuristics attempt to find improvements by making adjustments to the aggregate schedule without solving the entire model. For example, swapping heuristics could swap one or more duties among employees to find a better schedule. Many more improvement heuristics exist and it should be investigated what heuristic fits the situation.

10. Bibliography

- Bard, J. F., & Purnomo, H. W. (2005). Preference scheduling for nurses using column generation. *European Journal of Operational Research*, 164(2), 510-534.
- Brucker, P., Qu, R., & Burke, E. (2011). Personnel scheduling: Models and complexity. *European Journal of Operational Research*, 210(3), 467-473.
- Brucker, P., Qu, R., Burke, E., & Post, G. (2005). A decomposition, construction and post-processing approach for nurse rostering.
- Burke, E. K., De Causmaecker, P., Vanden Berghe, G., & Van Landeghem, H. (2004). The state of the art of nurse rostering. *Journal of scheduling*, 7(6), 441-449.
- Chan, P., & Weil, G. (2001). Cyclical staff scheduling using constraint logic programming. In E. Burke, & W. Erben, *Practice and Theory of Automated Timetabling III. PATAT 2000. Lecture Notes in Computer Science*, vol 2079. (pp. 159-175). Springer, Berlin, Heidelberg.
- De Nederlandse Veiligheidsbranche. (2017). *Caio Particuliere Beveiliging [PDF]*. Retrieved from <https://www.beveiligingsbranche.nl/caio/>
- Ernst, A. T., Jiang, H., Krishnamoorthy, M., & Sier, D. (2004). Staff scheduling and rostering: A review of applications, methods and models. *European journal of operational research*, 153(1), 3-27.
- Firat, M., & Hurkens, C. (2012). An improved MIP-based approach for a multi-skill workforce scheduling problem. *Journal of Scheduling*, 15(3), 363-380. doi:10.1007/s10951-011-0245-x
- Firat, M., Briskorn, D., & Laugier, A. (2016). A Branch-and-Price algorithm for stable workforce assignments with hierarchical skills. *European Journal of Operational Research*, 251(2), 676-685.
- Kasirzadeh, A., Saddoune, M., & Soumis, F. (2017). Airline crew scheduling: Models, algorithms, and data sets. *Euro Journal on Transportation and Logistics*, 6(2), 111-137. doi:10.1007/s13676-015-0080-x
- Kroon, L., & Fischetti, M. (2000). Scheduling train drivers and guards: the Dutch "Noord-Oost" case. In *33rd Annual Hawaii International Conference on System Sciences* (pp. 1-10).

- Maenhout, B., & Vanhoucke, M. (2013). An integrated nurse staffing and scheduling analysis for longer-term nursing staff allocation problems. *Omega*, 41(2), 485-499.
- Marasco, A., & Romano, A. (2011). A mathematical model for the management of a Service Center. *Mathematical and Computer Modelling*, 53(9-10), 2005-2014.
- Naudin, É., Chan, P. Y., Hiroux, M., Zemmouri, T., & Weil, G. (2012). Analysis of three mathematical models of the Staff Rostering Problem. *Journal of Scheduling*, 15(1), 23-38.
- Rijksoverheid. (2010). *The Working Hours Act [PDF]*. Retrieved from <https://www.rijksoverheid.nl/onderwerpen/werktijden/documenten/brochures/2010/05/10/de-arbeidstijdenwet-engels>
- Van den Bergh, J., Beliën, J., De Bruecker, P., Demeulemeester, E., & De Boeck, L. (2013). Personnel scheduling: A literature review. *European Journal of Operational Research*, 226(3), 367-385.
- Wan, L., & Bard, J. F. (2007). Weekly staff scheduling with workstation group restrictions. *Journal of the Operational Research Society*, 58(8), 1030-1046.

II. Appendix

II.1. Working Hours Act

		standard
Working time	per shift	12 hours
	per week	60 hours
	per week in a 4-week period	55 hours on average ¹⁾
	per week in a 16-week period	48 hours on average
Rest times	daily rest	11 hours (consecutive) (1 x per week; 8 hours, if necessary because of nature of work or business circumstances)
	weekly rest	36 hours (consecutive), or 72 hours per 14 days (split into periods of at least 32 hours)
Breaks	in the event of > 5.5 hours per shift	30 minutes (possibly 2 x 15 minutes)
	in the event of > 10 hours per shift	45 minutes (possibly 3 x 15 minutes)
	in the event of > 5.5 hours per shift	15 minutes ¹⁾
Sunday rest	Sunday work	no work on Sunday, except if: <ul style="list-style-type: none"> • in accordance with the type of work and stipulated or • necessary in connection with the type of work or business circumstances • agreed with works council (in absence of such: with employees involved) • individual consent
	free Sundays	13 (per 52-week period) any other number ¹⁾ , provided: <ul style="list-style-type: none"> • individual consent if fewer than 13 free Sundays per year
Night work Night shift: > 1 hour of work between midnight and 6 am	working time per shift	10 hours 12 hours, provided: <ul style="list-style-type: none"> • shift is followed by 12 hours of rest • 5 x per 2 weeks • maximum 22 x in 52-week period
	working time per week	40 hours (per 16 weeks), if ≥ 16 night shifts per 16 weeks
	rest time after night shift <i>applies for night shifts ending after 2 am</i>	14 hours (1 x per week; 8 hours, if necessary in connection with type of work or business circumstances)
	rest time after ≥ 3 night shifts	46 hours
	maximum length of series <i>applies if at least one of the shifts in the series is a night shift</i>	7 or 8 ¹⁾
	maximum number <i>applies for night shifts that end after 2 am</i>	<ul style="list-style-type: none"> • 36 night shifts per 16 weeks, or • 140 night shifts per 52 weeks¹⁾, or • 38 hours between midnight and 6 am per 2 consecutive weeks
	On-call duty	<ul style="list-style-type: none"> • 14 days free of on-call duty per 4 weeks • 2x2 days per 4 weeks no on-call duty and no work • no on-call duty 11 hours before or 14 hours after a night shift
	working time per 24 hours	13 hours
	working time per week in the event of on-call duty at night <i>applies if on-call duty is assigned between the hours of midnight and 6 am 16 times or more in a 16-week period</i>	<ul style="list-style-type: none"> • average 40 hours (per 16 weeks), or • average 45 hours (per 16 weeks), provided: <ul style="list-style-type: none"> - there is an uninterrupted 8 hours of rest before starting the new shift (in the event of last call between midnight and 6 am), or - 8 hours of uninterrupted rest in the 18 hours following 6 am (if the last call took place between midnight and 6 am and was immediately followed by a new shift).

11.2. Column generation solutions A

Run	A1	A2	A3	A4	A5	A6	A7	A8
Columns added	2930	2380	1769	1458	1342	1473	1053	667
Time (s)	700	494	598	522	396	375	227	70
Iteration 1	5210000018544	29427324682	66181535856	104047622419	454290212874	492950692811	301059539346	261351358622
2	4330000022459	5357167794	42000019462	50000017224	238007340314	234715979158	200701437263	194740376307
3	2831050621996	2307716360	40000019091	50000016854	141753531936	117607729940	150512843763	161635563746
4	1579682986913	1136387168	40000018892	50000016566	89636289443	89209992196	132355670845	141704044080
5	913067682048	1083356363	40000018737	50000016436	64485176879	82733345745	130000010002	135235263649
6	522708880988	869587762	40000018617	40000016354	60000014543	77171916067	130000009648	133333341619
7	268437317899	222244423	40000018498	40000016247	60000014009	72745370032	130000009499	130308590925
8	120637928320	21916.81	40000018395	40000016140	60000013744	71466837111	130000009416	130000007985
9	33092730246	21651.43	40000018285	40000016071	60000013577	70000011608	110000009293	130000007869
10	7832825171	21406.38	40000018160	40000015986	60000013385	70000011478	110000009220	130000007787
11	4285742420	21221.33	40000018049	40000015923	60000013280	70000011378	110000009177	130000007755
12	4285742420	21221.33	40000018049	40000015923	60000013280	70000011378	110000009177	130000007689
13	4285742420	21221.33	40000018049	40000015923	60000013280	70000011262	110000009144	130000007644
14	4285742420	21221.33	40000018049	40000015923	60000013280	70000011169	110000009113	130000007602
15	4285742420	21221.33	40000018049	40000015923	60000013280	70000011092	110000009079	130000007562
16	4285742420	21221.33	40000018049	40000015923	60000013280	70000011025	110000009066	130000007523
17	4285742420	21221.33	40000018049	40000015923	60000013215	70000010965	110000009046	130000007492
18	4285742420	21221.33	40000018049	40000015872	60000013215	70000010965	110000009032	130000007470
19	4285742420	21221.33	40000018049	40000015872	60000013134	70000010925	110000008998	130000007464
20	4285742420	21221.33	40000017964	40000015823	60000013080	70000010857	110000008974	130000007456
21	4285741623	21221.33	40000017964	40000015785	60000013041	70000010804	110000008960	130000007439

22	4285741623	21119.68	40000017917	40000015790	60000013041	70000010764	110000008937	130000007417
23		21019.35	40000017863	40000015790		70000010723	110000008929	130000007404
24			40000017815			70000010698	110000008924	130000007398
25			40000017815			70000010683	110000008920	130000007398
26						70000010667	110000008912	
27						70000010645	110000008907	
28						70000010645	110000008900	
29							110000008895	
30							110000008880	
31							110000008870	
32							110000008862	
33							110000008840	
34							110000008834	
35							110000008829	
36							110000008829	

Figure 25, Extended results column generation run A1-A8

11.3. Column generation solutions B

Run		B1	B2	B3	B4	B5	B6	B7	B8
Columns added		2919	2164	1333	1485	1485	1067	1284	648
Time (s)		473	358	352	416	317	211	148	57
Iteration	1	50000000824	17708.63	14854.11	12606469022	91720840318	86917656546	210000006338	50000004739
	2	10000019122	17609.51	14833.94	12647.99	43685780600	49083241939	154079639534	50000004739
	3	9000019276	17550.92	14803.83	12572.86	9321723921	18582153574	96243575466	50000004739
	4	8500019913	17505.90	14783.79	12526.35	11153.95	153727997	56300562324	50000004739
	5	4884525193	17466.74	14762.49	12501.52	10876.93	8842.14	30714175951	50000004739
	6	3293203027	17417.80	14741.50	12480.38	10782.34	8722.61	22876180779	37741941018
	7	704463435	17372.60	14726.44	12462.49	10741.48	8617.07	20000009398	32000005457
	8	462984910	17336.99	14711.61	12449.96	10709.33	8549.90	20000008701	30000005600
	9	21014.00	17307.48	14699.25	12438.55	10667.06	8491.77	20000008424	30000005556
	10	20625.00	17279.60	14687.10	12421.07	10642.76	8465.57	20000008225	30000005423
	11	20366.00	17240.93	14677.10	12411.90	10621.17	8449.44	20000008082	30000005191
	12	20366.00	17240.93	14677.10	12411.90	10621.17	8449.44	20000007844	30000005134
	13	20366.00	17200.63	14668.40	12399.23	10599.72	8421.59	20000007690	30000005131
	14	20366.00	17200.63	14659.01	12392.76	10581.13	8407.36	20000007690	30000005127
	15	20366.00	17170.81	14648.26	12385.33	10565.37	8383.38	20000007541	30000005097
	16	20366.00	17134.19	14638.44	12376.64	10541.96	8368.57	20000007488	30000005094
	17	20366.00	17134.19	14624.58	12371.07	10518.85	8349.40	20000007405	30000005079
	18	20366.00	17101.49	14616.17	12363.88	10506.94	8341.41	20000007311	30000005069
	19	20189.00	17076.40	14616.17	12353.80	10487.66	8328.75	20000007263	30000005065
	20	20189.00	17057.43	14610.67	12351.26	10487.66	8312.81	20000007227	30000005064
	21	20069.00	17057.43	14604.23	12351.26	10476.87	8299.86	20000007184	30000005061
	22	19974.00	17037.41	14600.54	12345.88	10469.96	8287.39	20000007155	30000005051

23	19974.00	17012.54	14596.09	12340.97	10462.91	8276.77	20000007096	30000005051
24		16994.70	14588.05	12337.73	10456.04	8270.35	20000007071	
25		16994.70	14581.85	12333.41	10451.15	8259.22	20000007045	
26			14576.67	12325.87	10447.45	8259.22	20000007008	
27			14573.30	12322.49	10440.59		20000006987	
28			14573.30	12317.31	10436.24		20000006956	
29				12309.44	10431.78		20000006948	
30				12304.96	10427.94		20000006930	
31				12301.32	10423.29		20000006913	
32				12297.47	10418.08		20000006913	
33				12295.19	10415.76			
34				12293.64	10410.77			
35				12293.64	10402.07			
36					10402.07			

Figure 26, Extended results column generation run B1-B8