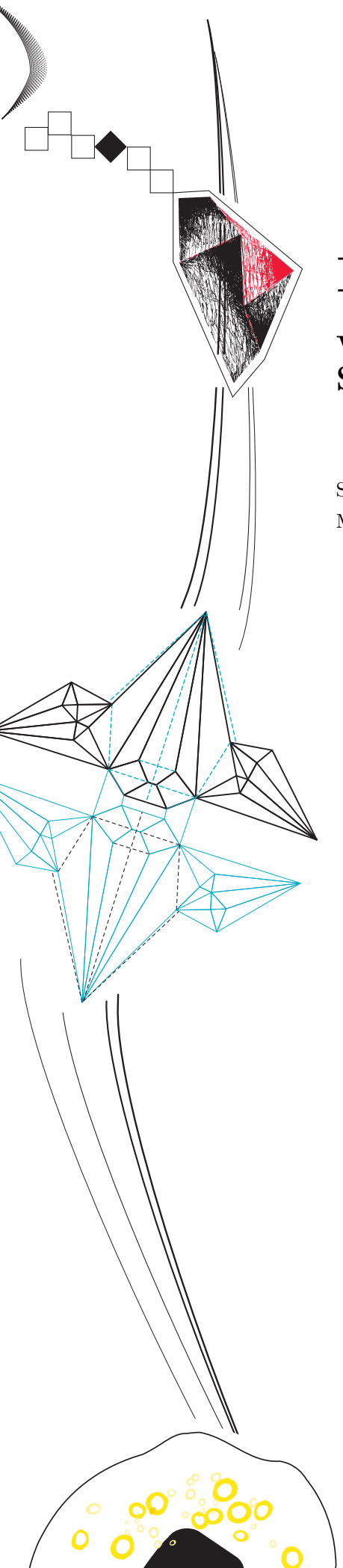


December 2, 2019

PREFERENCE LEARNING

WHAT DEFINES AN OPTIMAL SHIFT SCHEDULE?

Sanne van Weersel

Master thesis

Supervisors university

dr. ir. W.J.A. van Heeswijk

dr. ir. J.M.J. Schutten

University of Twente

The Netherlands

Company supervisor

L.M. Fijn van Draat, MSc

ORTEC

The Netherlands

UNIVERSITY OF TWENTE.

Management summary

Problem description

This research studies the problem of creating shift schedules. Since shift schedules are subjected to many strictly regulated rules, shift scheduling is considered to be a complex problem. Moreover, nowadays, preferences of employees are getting more and more important.

For this, ORTEC provides an optimization algorithm that creates shift schedules: the Optimizer. Using the Optimizer to create shift schedules brings several advantages, among others, time savings for the planners and an increased resource efficiency.

In the Optimizer rules are regulated as *hard constraints* and preferences as *soft constraints*. Practice has shown that translating scheduling preferences into soft constraints for the Optimizer is challenging and time consuming for the following reasons: i) It is hard to articulate preferences, ii) Preferences are company specific and iii) Preferences change over time. As a consequence, a vast amount of the current customers creates shift schedule manually and does not use the Optimizer.

Planners perceive these manually created shift schedules as desirable. Therefore, we expect that historical shift schedules contain valuable information that can be used to identify scheduling preferences of customers. The objective of this research is to explore how ORTEC can use these historical shift schedules to identify scheduling preferences and formulate soft constraints for the Optimizer.

Approach

This thesis focuses on identifying scheduling preferences for departments within an organization. Based on three customer cases, we formulate four categories of soft constraints:

- *Consecutive duties*: these constraints define if scheduling two duties on succeeding days is desirable, undesirable or indifferent.
- *Free weekends*: the preferred number of free weekends an employee should have per period. In this constraint, the customer is also required to define the length of the period in weeks for which the constraint holds.
- *Preferred series length*: the preferred minimum and maximum series length per type of contract. With a series length, we mean the number of consecutive shifts.

- *Distribution of weekend shifts*: This category contains two types of soft constraints: i) The preferred minimum and maximum series length of night shifts and ii) The frequency that employees should work night shifts.

For each of the types of soft constraints, we use both domain knowledge (e.g. information such as employee skills, demand for shifts, known preferences and labor laws) and the historical shift schedule to identify the preferences of the customer. Using domain knowledge helps us to structure the soft constraints and prevents us from suggesting irrelevant constraints. Thereafter, data mining techniques (e.g. cluster analysis and association rule mining) identify frequent patterns in the historical shift schedule of customers. These frequent patterns represent the scheduling preferences of the customers.

Results

For each of the categories of soft constraints, applying the method we propose gives the following insights:

- *Consecutive duties*: The algorithm detects 40% to 100% of the most important desirable and undesirable combinations of consecutive duties. Not all combinations detected by the algorithm are indeed a preference of the customer. Domain knowledge removes irrelevant combinations of duties and thereby improves the performance of the algorithm. In practice, the consultant and planner should verify whether all identified combinations of desirable and undesirable duties are relevant and if any imported combinations are missing.

- *Free weekends*: The algorithm is able to identify the preferred number of free weekends with a small error. However, the algorithm did not succeed in identifying the length of the period the constraint holds for. Domain knowledge ensures that the correct employees are included in the analysis and increases the performance of the algorithm. In practice, the consultants can trust the number of free weekends identified by the algorithm. However, the consultants should verify with the planner for which period the constraint holds.

- *Preferred series length*: The algorithm could identify the range of series length that occurred frequently in the past per type of contract. In most customer cases, this range was in line with the preferences of the customers. The algorithm detects frequent series lengths in the historical data, however, it did not succeed in identifying if the preference exists or not for all types of contracts. Therefore, in practice, consultants should verify with the planner if the preference exists or not.

- *Distribution of night shifts*: Using historical data, the algorithm could identify the preferred series length of night shifts for 4 out of 5 customer cases. Using domain knowledge, improves the performance of the algorithm. Moreover, we recommend to identify the frequency that such a series of night shifts should take place based on the demand for night shifts, the number of qualified employees and the personal preference of employees to work night shifts.

Conclusions

The method we propose gives insights in which events occurred frequently in the past. In this research, most customers perceived these frequent patterns as preferred and we could identify the preferences with only a small error.

This information supports the consultant and the planner during the implementation of the soft constraints in the Optimizer. We recommend ORTEC to use our method to identify scheduling preferences at existing customers. The results of our method can be used as a first version of the soft constraints for the existing customers. Next, feedback of the planner is required to fine-tune the results. In conclusion, we recommend ORTEC to use historical schedules and domain knowledge to simplify the process of incorporating preferences in the Optimizer.

Acknowledgments

With this thesis, I finish my master Industrial Engineering and Management and with that my time as a student at the University of Twente comes to an end. The past years have been a great time of learning new things and building strong friendships.

I would like to express my appreciation to ORTEC for giving me the opportunity to write this master thesis. In particular, I would like to thank Laurens Fijn van Draat for sharing his knowledge, having a critical look during the whole process and challenging me to achieve the best results. Additionally, I would like to thank Egbert van der Veen for the monthly brainstorm sessions, this really helped me in choosing the right direction of the research. Finally, I would like to thank my colleagues from OWS for making it a great work place, both working as a graduate intern and as a student assistant.

Furthermore, my acknowledgments go to Wouter van Heeswijk and Marco Schutten, my supervisors of the University of Twente, for providing useful feedback, giving valuable insights and being good discussion partners. Without their support, the whole process would have been much more complicated.

Last but not least, I would like to thank my friends, roommates and family for their input, feedback or support in any way.

Contents

Abbreviations

CLA Collective Labor Agreement.

GA Genetic Algorithm.

GI Greedy Insertion.

IQR Interquartile Range.

MAE Mean Absolute Error.

MSE Mean Squared Error.

NSP Nurse Scheduling Problem.

OWS ORTEC Workforce Scheduling.

RMSE Root Mean Squared Error.

SS Sum of Squares.

WHA Working Hours Act.

Chapter 1

Introduction

This research takes place at ORTEC in Zoetermeer. ORTEC provides software solutions to improve their customers' decision-making processes and operations. ORTEC provides these software solutions for different business processes, such as workforce scheduling, routing, loading, warehousing and field services. One of the products ORTEC provides is ORTEC Workforce Scheduling (OWS), which supports organizations with their shift scheduling. This chapter provides an introduction to this research. Section 1.1 describes the shift scheduling problem in general, followed by a description of the functionalities of OWS. Section 1.2 introduces the problem to tackle, followed by the objective of this research in Section 1.3. Section 1.4 describes the research framework.

1.1 Problem background

The shift scheduling problem is a well-studied problem and several approaches exist in the literature to solve this problem. This section provides a short description of existing approaches to solve the shift scheduling problem in Section 1.1.1. Section 1.1.2 provides an introduction of ORTEC's software solution for shift scheduling: OWS.

1.1.1 Shift scheduling

In the past few decades, shift scheduling has been heavily investigated within the fields of operations research and artificial intelligence (Van Den Bergh et al., 2013). According to Van Den Bergh et al. (2013), this increased research attention could be motivated by economic considerations of organizations. For many organizations, their workforce is one of their most valuable assets but also a major direct cost (Van Den Bergh et al., 2013). In other words, reducing the costs by a small percentage could already be very beneficial (El Adoly et al., 2018).

Shift schedules are subjected to many strictly regulated rules and company-specific rules. Consequently, the shift scheduling problem is to be considered as a complex process. An example of such a rule is the following Working Hours Act (WHA) rule: *'If a night shift ends after 2 am, this must be followed by a minimum of 14 hours of non-work time. This may be shortened to 8 hours a maximum of once per week. But only if the type of work or the business circumstances necessitate this.'* (Ministerie van Sociale Zaken en Werkgelegenheid, 2010). Additionally,

employees may have preferences for the shift schedule; for example, the preference to work at a specific location. Nowadays, it is getting more important for companies to satisfy employees, therefore, personal preferences are considered as well while creating a shift schedule (Van Den Bergh et al., 2013). Similarly, organizations have preferences while scheduling shifts; for example, assigning unpopular shifts fairly. For planners, it is hard to come up with a shift schedule that respects all labor laws and preferably satisfies the employees' and the organization's preferences (Brooks and Swailes, 2002). In the literature several approaches to solve the shift scheduling problem considering both labor laws and scheduling preferences exist; some of these approaches are:

i **Nurse Scheduling Problem**
Within operations research, the shift scheduling problem is also known as the Nurse Scheduling Problem (NSP). In the NSP, regulations and labor laws are modeled as *hard constraints*. To be considered feasible, a solution must satisfy all these hard constraints. Scheduling preferences are modeled as *soft constraints*. Satisfying these soft constraints increases the quality of a schedule, however, violation of a soft constraint can still lead to a feasible shift schedule (Blöchliger, 2004). Each soft constraint has an associated *weight* expressing its relative importance. The objective function of the optimization problem is two-fold. One part of the objective function minimizes the workforce costs incurred with a shift schedule. The other part represents to what extent the shift schedule satisfies the scheduling preferences, which is a combination between the weights and extent to which soft constraints are violated (Smet et al., 2013). Several methods exist to solve the NSP, for example, by using a linear programming model (Jaumard et al., 1998). The NSP is known to be an NP-hard problem (Solos et al., 2013). Consequently, small scheduling problems can be solved optimally, while complex scheduling problems are often solved by using a combination of heuristics (Brucker et al., 2011).

ii **Self-rostering**
Another approach to solve the shift scheduling problem is self-rostering. With self-rostering, employees propose a preferred shift schedule for the scheduling period (Van der Veen et al., 2016). In this way, self-rostering copes with the personal preferences of employees. In order to satisfy all hard constraints, the proposed shift schedules must comply with labor laws and the employee's contract hours. When the proposed schedules do not match the demand for shifts as defined by the organizations, shifts need to be reassigned. Reassigning these shifts can be solved by using an iterative improvement heuristic (Van der Veen et al., 2016). A drawback of self-rostering is that it only works under certain circumstances; a certain level of collegiality is required and the department should not be too large (Drouin and Potter, 2005). Moreover, Silvestro and Silvestro (2000) show that using a self-rostering system is difficult in departments that include more than 35 employees.

iii **Preference rostering**
Furthermore, the shift scheduling problem can be solved by preference rostering. In preference rostering, employees indicate their preference for the available shifts. De Grano et al. (2009) propose an auction model where employees bid for shifts and days off. Mathematical models are used to optimize the shift schedules of individual employees based on their bids

while satisfying all hard constraints. To be implemented successfully, the department size should not be too large and a certain level of collegiality should exist.

1.1.2 ORTEC Workforce scheduling

ORTEC provides a software solution, OWS, that supports organizations with creating shift schedules. In general, organizations are divided into different departments. For each department [1], the shift schedule is represented in a planning board as shown in Figure 1.1. The planning board displays all information needed to create a shift schedule for a given scheduling period. Table 1.1 describes the terminology used in the OWS planning board.

Table 1.1: Terminology OWS planning board

Scheduling period	The scheduling period specifies the period for which a schedule is created. The scheduling period is defined by a start- and end date.
Historical schedule	The shift schedule of the previous scheduling period.
Employees and contract hours	The employees and their contracted hours per week for this department.
Vacant shifts	A set of shifts required to satisfy demand during the scheduling period .
Total score	The total score adds up the violations of the soft constraints for all employees in the department. In this way, the total score represents the quality of the entire schedule.
Violations & contract hours	This section summarizes the following information per employee: i) The number of hard constraints violated, ii) The total score and iii) The difference between the number of hours worked and the total contracted hours in the scheduling period.

Planners can use OWS for manual-scheduling by assigning a vacant shift to one of the available employees. OWS supports the planner to cope with the high number of hard and soft constraints by highlighting violations and presenting the total score of the schedule. In addition, OWS has a feature to automatically create shift schedules for a department. Within OWS, this feature is known as the Optimizer. In short, the Optimizer assigns vacant shifts to the employees, resulting in a shift schedule.

In the Optimizer, the scheduling problem is formulated as an NSP. In order to use the Optimizer, all regulations and preferences must be explicitly modeled as a hard or a soft constraint. Figures 1.2 and 1.3 show an example of one of the hard and soft constraints, respectively. The white input fields allow the customer to change the parameters of a constraint. In contrast to hard constraints, soft constraints require a weight between 1 and 10,000, which reflects its relative importance. Currently, OWS contains around 110 types of hard constraints and 40 types of soft constraints. In order to use the Optimizer, customers select a set of relevant hard and soft

[1] In practice, a department is often split up into several scheduling groups. In this research we refer to a scheduling group as a department.

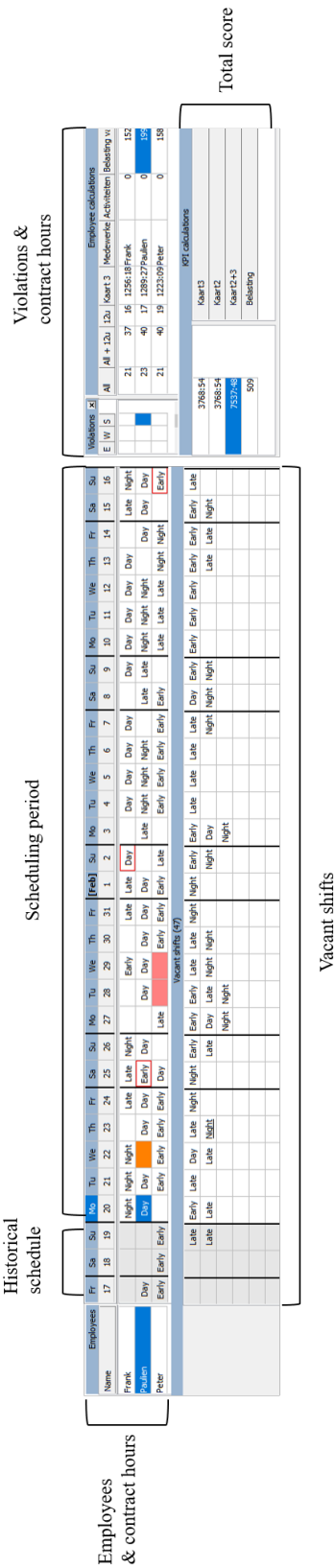


Figure 1.1: ORTEC Workforce Scheduling planning board

constraints that represent their scheduling problem. Subsequently, each of these hard and soft constraint can be customized by changing its parameters and, if needed, its weight.

The scheduling problems of ORTEC's customers are often complex problems; shift schedules for large departments must be created considering a large number of constraints. For one of the customers, every week 1952 shifts have to be scheduled in a department including 139 employees. Due to the complexity of these scheduling problems, the Optimizer uses a combination of several heuristics to solve this optimization problem within a reasonable time. Chapter 2 provides a detailed description of the heuristics used in the Optimizer.

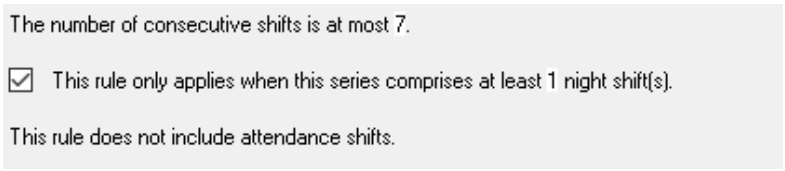


Figure 1.2: Example of a hard constraint in OWS

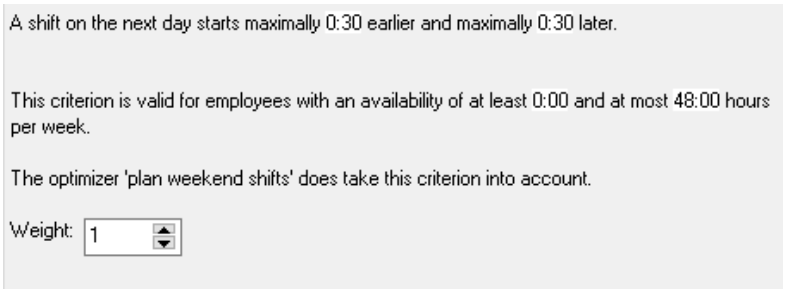


Figure 1.3: Example of a soft constraint in OWS

1.2 Problem statement

Compared to manual-scheduling, using an optimization algorithm to solve the shift scheduling problem brings several advantages for organizations. Among other things, the following benefits are known in the literature and from ORTEC's practice:

i **Resource efficiency**
In general, optimization algorithms create more efficient shift schedules compared to manual-scheduling performed by planners resulting in an increased resource efficiency (Burke et al., 2004; El Adoly et al., 2018). To be more specific, the Optimizer has the potential to reduce the workforce costs with 2% as a result of more efficient shift schedules (Prof. Dr. G. Kant, personal communication, September 30, 2019).

ii **Less administrative workload**
Especially for large organizations, scheduling is a time-consuming task. Using an optimization algorithm for the shift scheduling problem reduces the administrative workload for planners (Burke et al., 2004; El Adoly et al., 2018). For one of ORTEC's customers, using the Optimizer reduced the time spent on planning and administrative workload with 2 FTEs (Dr. E. van der Veen, personal communication, May 20, 2019).

iii **Less understaffing**

A shift schedule is called understaffed if not all vacant shift could be scheduled in a scheduling period. In health care, for example, a tense labor market exists. Especially in these kinds of industries, it is important to create efficient shift schedules to prevent understaffing. With manual-scheduling it is challenging to schedule all vacant shifts without violating any of the hard constraints. In general, using an optimization algorithm results in less understaffing compared to manual-scheduling. This benefit was proved in practice: implementing the Optimizer resulted in more efficient shift schedules and thereby solved the problem of understaffing in a Dutch academic hospital (Prof. Dr. G. Kant, personal communication, September 30, 2019).

iv **Improved employee satisfaction**

For planners, it is difficult to create good and fair schedules while considering all labor laws, regulations and preferences (Brooks and Swailes, 2002). Optimization algorithms are able to deal with employee's preferences while at the same time respect all regulations. Therefore, using an optimization algorithm to solve the shift scheduling problem could improve employee satisfaction (Burke et al., 2004; El Adoly et al., 2018). In a Dutch hospital, employee satisfaction research was conducted before and after implementing the Optimizer. One of the measurements in the survey was work-life balance, the balance between time at work and leisure time. In the hospital, the work-life balance improved from 4/10 to 6.7/10 after implementing the Optimizer (Prof. Dr. G. Kant, personal communication, September 30, 2019).

Summarizing, using the Optimizer brings several advantages for organizations. However, before an optimization algorithm can be used to create shift schedules, the objective function and constraints must be explicitly modeled. In other words, real-life requirements and preferences must be translated into a mathematical formulation. In contrast to preferences, rules are defined precisely and therefore translatable into mathematical hard constraints. However, translating scheduling preferences to mathematical soft constraints is challenging and time-consuming for several reasons:

i **It is hard to articulate preferences**

Planners use complex decision-making skills to create a shift schedule (Kellogg and Walczak, 2007). Scheduling preferences are often known from years of experience. Consequently, planners find it hard to articulate these preferences. Often multiple improvement iterations are required to formulate a complete set of constraints including their parameters and weights reflecting the customers' preferences. Therefore, incorporating soft constraints in the Optimizer is a time-consuming process for both consultant and customer.

ii **Preferences are company specific**

Preferences are personal; therefore, different soft constraints are found between sectors, countries and organizations (Smet et al., 2013). Consequently, ORTEC's consultants must define the set of constraints for each customer separately. Since every department has its own agreements among colleagues, the preferences may even differ within an organization.

iii **Preferences change over time**

Finally, personal preferences may change over time. For example, employees who preferred to work during the weekend in the past might prefer to work weekday shifts nowadays. Additionally, the scheduling problem of a department may change over time, for example, in the case new employees join the company. Consequently, the set of constraints holds only for a certain period. Therefore, soft constraints need to be updated if the preferences or the scheduling problem within an organization or department change.

In conclusion, incorporating preferences in an optimization algorithm is a challenging and time-consuming process. As a consequence, a vast amount of customers does not use the Optimizer. ORTEC aims to reduce the time and effort needed to formulate soft constraints. In this way, ORTEC hopes that in the future more customers can benefit from the full potential of OWS by using the Optimizer.

1.3 Research objective

According to ORTEC customers, manually-scheduled shift schedules are perceived as desirable schedules as these include implicit preferences. For this reason, we expect that these historical manually-scheduled shift schedules contain valuable information that can be used to formulate soft constraints for the Optimizer. Earlier research within ORTEC explored the possibilities of identifying soft and hard constraints in historical schedules (Hassan, 2019). This research had a theoretical focus; artificial shift schedules were created and several machine learning and data mining techniques have been applied to identify interesting patterns in these artificial schedules. This research proved that it is possible to extract interesting patterns from shift schedules. The next step for ORTEC is to analyze historical schedules from real-world cases. In addition to identifying interesting patterns in historical schedules, ORTEC aims to use their domain knowledge in understanding the scheduling preferences of their customers. The idea is that by bringing in domain knowledge, the performance of data mining techniques improves.

Summarizing, ORTEC's objective is to explore how historical schedules and domain knowledge can be used to identify scheduling preferences. We can divide this objective into three parts: i) Formulate relevant types of soft constraints, ii) Identify the parameters of these relevant soft constraint and iii) Determine the relative weight for each of these soft constraints. In this research, the focus lies on part one and two. First, we identify what type of soft constraints are required to model the scheduling preferences of ORTEC's customers. Secondly, we identify the parameters of these relevant soft constraints. Due to time limitations, learning the relative weights of these soft constraints is out of scope. The research question is formulated as follows:

"How can ORTEC use historical manually-scheduled schedules and domain knowledge to formulate customer-specific soft constraints for the Optimizer?"

1.4 Research framework

In order to answer the research question, we define four sub-research questions. In Chapter 2, we describe the Optimizer of OWS. This includes a description of the types of constraints currently used in OWS, the optimization algorithm that is applied in OWS and the scheduling preferences that typically exists at ORTEC's customers. This results in the following research questions:

1 *How is the Optimizer currently used in OWS?*

   (a) *Which types of constraints can we distinguish in OWS?*

   (b) *What optimization algorithm is applied in the OWS Optimizer?*

   (c) *Which type of soft constraints are used at ORTEC's customers?*

Chapter 3 provides an overview of what has been done in the literature to learn soft constraints based on historical data. Thereafter, we describe data mining techniques that can be used to identify preferences in data sets. Finally, after identifying the soft constraints based on historical data, it is interesting to evaluate the performance of the applied method. For that reason, we describe possible evaluation metrics to evaluate the performance of data mining techniques. This results in the second set of research questions:

2 *What can we learn from the literature about identifying preferences from data?*

   (a) *What techniques exists to learn soft constraints from historical data?*

   (b) *What data mining techniques can be used to identify preferences from data?*

   (c) *What evaluation metrics are available to determine the performance of a data mining technique?*

In Chapter 4, we describe the approach to identify soft constraints using historical shift schedules. In this chapter, we structure the problem, propose how to use domain knowledge and describe what data mining techniques can identify the parameters of the soft constraints. Finally, this chapter describes how to evaluate the performance of the method we propose. Resulting in the following set of research questions:

3 *How can we identify the parameters of relevant soft constraints?*

   (a) *What set of soft constraints represents the scheduling preferences of ORTEC's customers?*

   (b) *What domain knowledge can be used to identify the scheduling preferences of the customers?*

   (c) *Which methods are able to identify the parameters of each of the types of soft constraints?*

   (d) *How can we evaluate the identified parameters for each of the types of soft constraints?*

Consequently, we apply the approach to each of the soft constraints and evaluate the results. Chapter 5 describes the results of the method applied for each of the soft constraints. First, we evaluate to what extent the parameters of the soft constraints could be identified using historical shift schedules and domain knowledge.

4 *For each of the formulated soft constraints, to what extent can the parameters be identified based on historical data and domain knowledge?*

Answering these sub-research questions gives insight into how ORTEC can use historical schedules and domain knowledge to identify relevant soft constraints for a customer. This thesis ends with a conclusion and recommendations for ORTEC in Chapter 6.

Chapter 2

Problem context

The problem of this research is defined as follows: it is challenging to incorporate scheduling preferences in an optimization algorithm such as the Optimizer. This chapter provides a description of the information required to understand the context of the problem. Section 2.1 provides a description of the data entities in the available historical shift schedules. Section 2.2 describes the constraint types used in OWS and the approach used by Optimizer to solve the scheduling problem. Section 2.3 provides a description of the types of soft constraints used at three customers. Section 2.4 provides an answer to the first research question: *How is the Optimizer currently used in OWS?*

2.1 Customer data

The historical data available contains the following data entities: i) Duties, ii) Shifts, iii) Duty Demand, iv) Employees, v) Historical shifts.

i **Duties**
Each customer's database includes a set of duties, D. A duty is defined by one or multiple tasks with a begin- and end time in hours on a day. In some organizations, a duty also includes a required skill level and location. It is important to notice that a duty does not include a date. In this thesis, duties are indicated by a, or b, where $a, b \in D$.

ii **Shift**
A shift consists of a duty, a date and an employee that works the shift.

iii **Duty demand**
The duty demand describes the number of shifts required with a specific duty for each day of the week. In this way, the duty demand is the same for all weeks. The planner can also change the duty demand for a specific week.

iv **Employee**
A department has a set of employees, E. An employee, e, has a contract with an end-date, contracted hours per week and set of skills.

v **Historical shift**

A historical shift is a shift that took place in the past. In this thesis, historical shifts are indicated by h.

vi **Historical shift schedule**

The historical shift schedule, H, is the set of all historical shifts h.

Figure 2.1 shows the relation between the data entities.

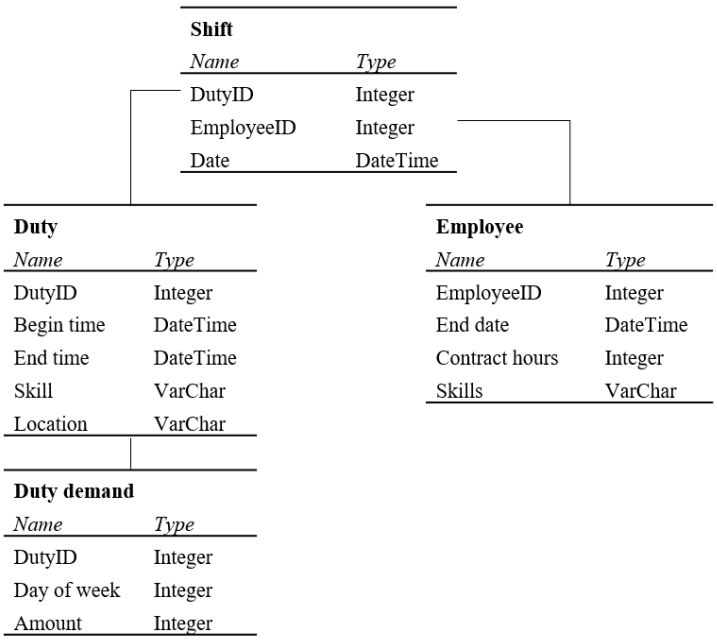


Figure 2.1: Relation between data entities used in this thesis

Data from eight customers cases from three different organizations have been collected. To ensure the anonymity of the customers, we refer to the customers using the following abbreviations: A1, B1, B2, B3, C1, C2, C3 and C4. These customers operate in different industries and countries: customer A is a Dutch academic hospital, customer B is a Finnish retail company and customer C is a Finnish home care provider. Table 2.1 shows a summary of the size of the scheduling problems of the customer cases. The size of the department differs per customer case. Additionally, the table represents the total contracted hours for each department. For each of these customer cases, at least 40 months of historical data is available.

Table 2.1: Description of the scheduling problems of the customer cases

Customer	Number of duties	Number of shifts per week	Number of employees	Total contracted hours (per week)	Length historical schedule (months)
A1	40	177	131	1795	60
B1	49	196	35	812	40
B2	36	112	40	1059	40
B3	65	93	30	732	40
C1	36	131	61	1395	48
C2	11	45	40	460	48
C3	43	161	76	1659	48
C4	36	131	62	1231	48

2.2 OWS Optimizer

This section introduces the categories of constraints used in the Optimizer, the way these constraints are modeled, and the algorithm that solves the shift scheduling problem.

2.2.1 Constraints in OWS

OWS contains three types of constraints: i) Hard constraints, ii) Soft constraints and iii) Employee-wishes. All these constraints can be formulated on different levels within the organization. The remaining of this section consists of a detailed description of each of the categories and the levels where the constraint can be formulated on.

i **Hard constraints**
Hard constraints can be formulated on the following four levels:

(a) *National level*
Hard constraints on national level represent labor laws that apply to all employees in a country. An example of this constraint is the following WHA rule:

- *'An employee must have at least 13 free Sundays per year.'* (Ministerie van Sociale Zaken en Werkgelegenheid, 2010).

(b) *Industry level*
Hard constraints on industry level contain industry-specific rules, in The Netherlands also known as the Collective Labor Agreement (CLA). An example of such an agreement:

- An employee has the right to have at least 26 free weekends a year.

(c) *Department level*
These constraints apply for all employees in a department, for example:

- An employee may work at most 3 night shifts in a row.

(d) *Individual level*
Hard constraints on individual level apply for a specific employee, such as:

- Fixed assignments, for example, an employee works every Monday evening.

- Personal agreements, for example, an employee is not required to work at location B.

ii **Soft constraints**

Soft constraints can be formulated on the following two levels:

(a) *Department level*

These soft constraints apply for all employees in a department, for example:

- A shift with duty type 1 should be followed by a shift with duty type 2.

(b) *Group level*

Constraints formulated on group level hold for a specific group of employees within a department, for example, part-time employees. For example:

- Part-time employees should work at most 2 weekend shifts a month.

iii **Employee-wishes**

Employee-wishes can either be modelled as a hard constraint or a soft constraint. Employees can submit a *request not to work* during a given period or a specific shift. Subsequently, the planner can decide if this wish should be modeled as a hard constraint or soft constraint. Additionally, employees can *request to work* during a given period or a specific shift. Since there is no guarantee that the preferred shift is available, a *request to work* is always modeled as a soft constraint. If the employee-wish is modeled as a soft constraint, planners must assign a weight to this employee-wish. Examples of employee-wishes are:

- An employee wishes a day-off on 01-01-2020.
- An employee wishes not to work on every other Monday.
- An employee wishes to work a night shift on Thursday.

In this research, the focus lies on identifying the parameters of soft constraints in historical shift schedules. Hard constraints and employee-wishes are out of scope for the following reasons: i) Hard constraints are derived from labor laws and collective agreements, and ii) Employee-wishes are requested by the employee. In other words, both hard constraints and employee-wishes are known, therefore there is no need to identify these constraint types based on historical data. However, as hard constraints define the boundaries of the soft constraints, we must keep the relation between the different categories of constraints in mind.

2.2.2 Model formulation

In OWS, a hierarchy is used to distinguish the different categories of constraints. Hard constraints have the highest priority; the Optimizer respects all formulated hard constraints while generating a shift schedule. Therefore, the optimizer will not schedule a vacant shift if this would result in a violation of one of the hard constraints. Furthermore, assigning all vacant shifts has the second-highest priority. If possible, the Optimizer always assigns an additional vacant shift to an employee, even though this results in a violation of a soft constraint. Finally, minimizing the objective function, and thus satisfying scheduling preferences, has the lowest priority.

The Optimizer includes a set of soft constraints. The Optimizer collects all violations of a soft constraint for the given scheduling period. We define these violations as a set V, where a violation $v \in V$. The terminology of the Optimizer is listed in Table 2.2.

Table 2.2: Terminology OWS optimizer

Decision variables	The decision variables determine whether a specific shift is assigned to a specific employee or not.
Deviation	The absolute extent to which a soft constraint is violated.
Severity	The penalty incurred when a soft constraint is violated, which is based on the extent of the deviation and cost type.
Cost types	In OWS two different cost types are possible: 1) linear, where the severity is equal to the deviation, and 2) quadratic, where the severity is equal to the squared deviation.
Weight	A weight between 1 and 10.000 is assigned to each of the soft constraints, which reflects the relative importance of satisfying this soft constraint.
Total score	The total score equals the severity of violating a soft constraint multiplied by its weight. This means that if the weight is zero, the soft constraint is disregarded.
The objective function	The objective function minimizes the total penalty obtained. In this way, the Optimizer is stimulated to satisfy the soft constraints while creating shift schedules. In addition to the penalties obtained, in OWS personnel costs are taken into account in the objective function. As monetary costs are not relevant in this research, these costs are left out in the objective function.

The objection function [1] of the Optimizer is to minimize the sum of all penalties obtained from violating soft constraints, i.e. the total score. Equation 2.1 shows the calculation for the *total score*. The height of the penalty incurred from violating a soft constraint depends on i) The severity of the deviation and ii) The weight of the soft constraint. The severity of a deviation depends on the extent of the deviation and the cost type of a constraint, see Equation 2.2. The extent to which a constraint is violated depends on the preferences as defined by the customer and the realized shift schedule. For example, if a soft constraint defines that employees should work at most 1 night shift per week and the employee works 3 night shifts in a specific week, the deviation is equal to 2. The cost type is predefined for each constraint type and can either be linear or quadratic. In the case a constraint has linear costs, the severity is equal to the deviation. In the case a constraint has quadratic costs, the severity is equal to the squared deviation. By using quadratic costs larger deviations from the preferences are considered to be more severe.

[1] As monetary costs are not relevant in this research, the objective function in this research is the minimization of the sum of all penalties obtained.

$$TotalScore = \sum_{v \in V} Severity_v * Weight_v \tag{2.1}$$

$$Severity = \begin{cases} Deviation, & \text{In case of linear costs} \\ Deviation^2, & \text{In case of quadratic costs} \end{cases} \tag{2.2}$$

2.2.3 Algorithm OWS

The Optimizer goes through the following four consecutive phases to generate a shift schedule: i) Greedy Insertion construction heuristic, ii) Genetic Algorithm, iii) Local optimization, iv) Ruin and recreate. A full description of the algorithm used in OWS can be found in (Post and Veltman, 2004).

1 **Greedy Insertion construction heuristic**
The objective of the first phase is to create a set of initial solutions. The user can define the number of initial solutions to create. First, all vacant shifts receive a score that represents the difficulty to schedule. This score is based on several criteria, for example, a weekend shift receives a higher score than a shift on a weekday. Thereafter, the list of vacant shifts is sorted in decreasing order of their score. A Greedy Insertion (GI) construction heuristic is used to assign every vacant shift to an employee in the sequence of the sorted list. To obtain several initial solutions, the GI is executed multiple times. In each iteration, the GI divides the employees over different subsets. Subsequently, as much as possible shifts are assigned to the employees in the first subset, then the remaining shifts are assigned to employees in the other subsets. By creating different subsets of employees, the GI can create different initial solutions.

2 **Genetic Algorithm**
Once the GI created a set of so-called parent solutions, the Genetic Algorithm (GA) will be initialized to generate children solutions. The GA uses five types of operators to generate new solutions: i) Two crossover operators, which combines two parent solutions into two child solutions, and ii) Three mutation operators, which transform one parent solution into one child solution. In each iteration, the GA randomly selects one of these five GA operators and applies this on one or two random parent solution(s). If the operator results in an infeasible child solution, a repair heuristic is applied to guarantee the feasibility of a solution. The children solutions are saved in a new set of solutions, where the GA is also applied. The GA continues with creating children solutions until the maximum computation time for global optimization is reached The best solution found so far is selected. The maximum computation time for global optimization can be defined by the user.

3 **Local optimization**
The algorithm continues with a local optimization phase to improve the current best solution. Four different types of moves are possible within the local optimization phase: 1-opt, 2-opt, 1-opt(k) and 2-opt(k). A 1-opt move removes a shift from an employee and assigns it to another available employee. A 2-opt move exchanges two shifts assigned to different

employees. When such a move results in a better solution, the new solution is accepted. Once a schedule cannot be improved anymore using 1-opt changes, the solution is 1-opt optimal and the algorithm starts a 2-opt search. If an improvement is found in the 2-opt search, the algorithm returns to the 1-opt search. When no improvements can be found for the 2-opt search, the 1-opt(k) search is initiated. In a 1-opt(k) search, k consecutive shifts are moved from one employee to another employee. Every time a better solution is found, the algorithm returns to the 1-opt search. The algorithm continues until it reaches 2-opt(3) optimality or when no improvements can be found within reasonable time.

4 **Ruin & recreate**
Finally, a ruin & recreate heuristic is applied to the best-found solution so far. By ruining and recreating the solution, the algorithm tries to escape from a local optimum. Two different operators exist to ruin the current solution. The first operator randomly selects a predefined number of employees using the roulette wheel principle: employees with a higher total penalty have a higher probability to be selected. All shifts within the scheduling period assigned to the selected employees will be removed. The second operator removes a predefined number of randomly selected shifts in the schedule. To recreate the solution, GI is applied to assign the vacant shifts to the available employees. One of the operators is randomly selected, where the probability is based on the success rate of previous iterations. The variable neighborhood search continues until the predefined total computation time is reached. The best-found solution is returned to the end-user and presented on the planning board.

2.3 Scheduling preferences

Data sets of eight departments from three different customers of ORTEC have been collected. As these customers currently use the Optimizer, their scheduling preferences and their corresponding soft constraints are known. In total seven different types of soft constraints are currently used by these customers. Table 2.3 provides an overview of these constraints and their description.

The first soft constraint type is *Avoid combinations*. For example, shifts with duty type E should not be scheduled after shifts with duty type L on succeeding days.

The second soft constraint type is *change in begin time*. For example, a shift on the next day starts maximally 0.5 hours earlier and 1 hour later.

The third soft constraint type, *same duty*, indicates that two consecutive shifts should have the same duty. Consecutive shifts are shifts that are worked on two succeeding days by the same employee.

The fourth soft constraint type, *weekends off*, regulates the number of free weekends for employees. In this research a free weekend is defined as follows: an employee has a free weekend if no work takes place between Saturday 04:00 and Sunday 23:59. A customer can indicate a preference for both the number of free weekends and the number of weeks the constraint should consider, for example:

- Constraint 1: An employee should have at least 2 free weekends in a period of 4 weeks
- Constraint 2: An employee should have at least 1 free weekend in a period of 2 weeks

Table 2.3: Categories of the soft constraints used at ORTEC's customers

Soft constraint	**Description**
Avoid combinations	Shifts with duty type *[DutyType1]* should not be scheduled after shifts with duty type *[DutyType2]* on succeeding days.
Change in begin time	A shift on the next day starts at most *[NHours earlier]* hours earlier and *[NHours later]* hours later
Same duty	Consecutive shifts should have the same duty
Weekends off	In a period of *[NWeeks]* weeks, an employee should have at least *[NWeekends]* weekends off.
Number of night shifts	Employees should work between *[Minimum]* and *[Maximum]* number of night shifts in a period of *[NWeeks] weeks.*
Preferred series length	A series of shifts should have a minimum length of *[Minimum length]* and a maximum length of *[Maximum length].* This constraint holds only for employees with a contract of minimal *[Minimum hours]* and maximal *[Maximum hours]* hours.
Preferred series length night shifts	A series of night shifts should have a minimum length of *[Minimum length]* and a maximum length of *[Maximum length].*

The average number of free weekends is the same in both examples, however, example 2 is more strictly formulated. Figure 2.2 shows an example of a shift schedule for four consecutive weekends. In this example, constraint 1 is satisfied during the whole scheduling period. However, constraint 2 is violated, as the employee works both weekends in week 2 and 3. This results in a deviation of 1. In OWS, this constraints has quadratic costs, meaning that the deviation is squared.

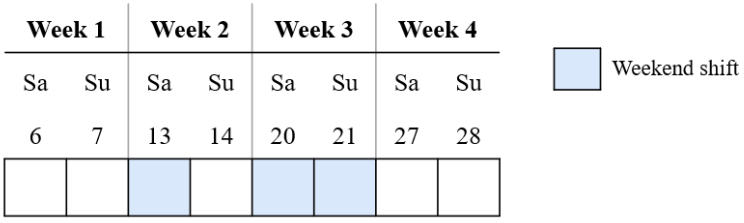


Figure 2.2: Example constraint free weekends

The fifth soft constraint type is *number of night shifts*, for example: An employee should work between 2 and 3 night shifts in a period of 3 weeks. Figure 2.3 shows an example shift schedule for three weeks. In this period of three weeks, the employee works 4 night shifts, consequently, the deviation of the soft constraint is equal to 1 ($= 4 - 3$). In OWS this constraint type also has quadratic costs.

The constraint type *preferred series length* indicates the boundaries of the preferred series

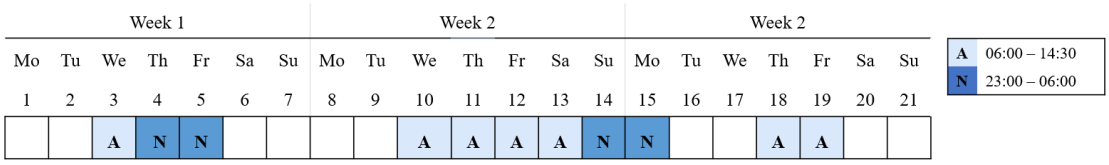


Figure 2.3: Example constraint number of shifts

length. A shift series includes shifts worked by the same employee on succeeding days. A series is interrupted if an employee does not work between 0:00 and 0:00 the next day. An example of such a constraint is: a series of shifts should have a minimum length of 2 and a maximum length of 3. Figure 2.4 shows an example of a shift schedule for two weeks including three shift series. In this example, the first shift series lies within the boundaries, the second shift series deviates 1 and the third shift series deviates 2. In OWS this constraint type has quadratic costs, therefore, the aforementioned example would result in a severity equal to 5 ($= 1^2 + 2^2$).

Finally, customers indicate a preference for the series length of night shifts, this constraints works the same as the constraint *Preferred series length*. However, this constraint type takes only consecutive night shifts are taken into account.

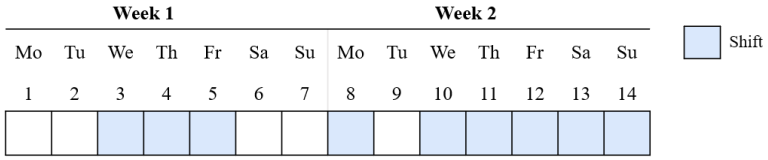


Figure 2.4: Example constraint preferred series length

Table 2.4 summarizes the number of soft constraints used at each of the customer cases. This table shows that for each customer a different set soft constraints is relevant.

Table 2.4: Summary of the number of soft constraints used per customer case

	Customer							
Soft constraint	**A1**	**B1**	**B2**	**B3**	**C1**	**C2**	**C3**	**C4**
Avoid combination	8	-	-	-	2	-	6	4
Change in begin time	-	-	-	-	1	1	1	1
Same duty	1	-	-	-	-	-	1	-
Weekends off	-	1	1	1	-	-	-	-
Number of night shifts	1	-	-	-	1	1	1	1
Preferred series length	-	1	1	1	2	2	2	2
Preferred series length of night shifts	1	-	-	-	1	1	1	1

2.4 Conclusion

This chapter describes the constraint types, the algorithm used in OWS and the soft constraints used at ORTEC's customers. With this information, we can answer the following sub-research question:

How is the Optimizer currently used in OWS?

In OWS three constraint types exist: i) Hard constraints, ii) Soft constraints and iii) Employee wishes. In this thesis, the focus lies on identifying soft constraints. Soft constraints can be formulated both on department and group level.

In OWS, shift schedules are created by the Optimizer. The Optimizer goes through four consecutive heuristics to solve this optimization problem. The objective of the Optimizer is to minimize the total score, which consists of the sum of penalties obtained from violating and satisfying soft constraints. ORTEC's customers typically use a combination of the following seven types of soft constraints:

- Avoid specific combinations of consecutive shifts
- Change in begin time of consecutive shifts
- Consecutive shifts should have the same duty
- The preferred number of weekends off
- The preferred number of night shifts
- The preferred series length
- The preferred series length for night shifts

Chapter 3

Literature review

The objective of this research is to identify the scheduling preferences of customers based on historical shift schedules. The previous chapter describes soft constraints representing the customers' scheduling preferences. The next step is to identify the parameters of these soft constraints. To achieve this, literature is used to explore what already has been done in the fields of constraint learning. Section 3.1 provides an overview of relevant literature. Additionally, Section 3.2 discusses data mining techniques, which can be applied to discover interesting patterns in data sets (Han et al., 2012a). Finally, Section 3.3 provides an overview of frequently used evaluation metrics in data mining that can be used to determine the performance of a data mining technique.

3.1 Constraint learning

In this section, we describe what has been done in the literature in the fields of constraint learning and use this as an inspiration for this research. Constraint learning refers to the problem of finding a set of constraints that is satisfied in a given data set (De Raedt et al., 2018). In other words, the constraints found should not be violated in the historical data set. First, we describe approaches to learn hard constraints from multi-dimensional data, such as the scheduling problem, followed by an overview of existing approaches to learn soft constraints.

In the literature, the following methods to learn hard constraints in multi-dimensional data exist. First, Beldiceanu and Simonis (2012) introduce ModelSeeker, an approach to learn hard constraints from historical data. ModelSeeker uses constraints that are defined in a global constraint catalogue, a list of frequently used constraints in the literature. ModelSeeker creates a ranked list of constraints that match the data. Using ModelSeeker, the hard constraints underlying the scheduling of the German Football Championship, the Bundesliga, could be identified (Beldiceanu and Simonis, 2012). Second, Kumar et al. (2018) propose COUNT-OR, an approach to learn hard constraints in scheduling problems. The input for COUNT-OR is a historical schedule and a given set of hard constraints. COUNT-OR changes the parameters of this set of hard constraints in such a way that the historical schedule does not violate these hard constraints. COUNT-OR can identify 11 types of hard constraints (Kumar et al., 2018). This method was applied to a scheduling problem with 49 nurses, a scheduling period of 28 days and 4 different

shift types. Compared to ORTEC's customers, where the number of shift types varies between 10 and 250 shift types, COUNT-OR was applied on a relative small scheduling problem. Both ModelSeeker and COUNT-OR aim to find the boundaries of hard constraints rather than soft constraints that represent the scheduling preferences. These methods set the parameters of the constraints equal to the extreme cases found in historical data, while in this research we aim to find frequently occurring patterns, therefore, these methods are not applicable in this research.

In contrast to learning hard constraints, relatively few approaches for learning soft constraints have been proposed in the literature. Approaches for learning soft constraints can be classified into two categories with different objectives: i) Learning the weights of the soft constraints and 2) Learning the structure of the soft constraints, where both the parameters as the weights of the constraints are being learned (De Raedt et al., 2018).

Rossi and Sperduti (2004) introduce an interactive framework to learn the weights of soft constraints. Within this framework, users rank constraints and solutions according to their preference, subsequently, these scores are used to learn the relative weights of the given constraints (Rossi and Sperduti, 2004). A similar kind of approach was proposed by Teso et al. (2016), where interactive learning is used to the weights of soft constraints in customization and design problems. In these two approaches, the set of relevant soft constraints including their parameters is given and only the relative importance of those soft constraints is being learned. In this research the parameters of the soft constraints are not known, therefore, we cannot use these approaches.

Campigotto et al. (2015) propose a method, CLEO, to learn the structure of soft constraints. In this method, all possible soft constraints are enumerated. Subsequently, a weight is assigned to the soft constraints using the objective function. The objective function maximizes the utility of a schedule, where the utility is the sum of the weights of the satisfied constraints. Moreover, the objective function aims to set the weight of most of the constraints to zero, resulting, all irrelevant constraints are eliminated. Finally, in an interactive refinement phase, the parameters and weights of the constraints are tuned by human feedback. CLEO is tested in a setting with at most 9 types of soft constraints to enumerate. As ORTEC's customers deal with complex scheduling problem, enumeration of all possible soft constraints would result in several hundreds of soft constraints. Moreover, CLEO requires human feedback during the refinement phase, while the objective of this research is to develop a method where no interaction with humans is required. To deal with a large number of constraints, Campigotto et al. (2011) suggest using known hard constraints to define the boundaries of the soft constraints to identify.

3.2 Data mining techniques

Data mining is a technique that can be applied to find interesting patterns in data sets. Han et al. (2012a) distinguish five types of data mining techniques to detect patterns in data sets: i) Data discrimination, ii) Frequent patterns, iii) Classification, iv) Cluster analysis and v) Outlier analysis.

The first type is *data discrimination*, which compares data of two or more different classes and describes the differences between those classes. The user defines a target class and a contrasting class, thereafter, a data technique evaluates the differences between these classes. The user is required to define the classes before the technique can be used. In this research, we aim to

distinguish desirable scheduling decisions from undesirable scheduling decisions, these would be the different classes in the data set. However, we do not know how these classes look like in advance. For this reason, data discrimination is no usable data technique in this thesis.

The second type is *frequent patterns and association mining*, which extracts frequent patterns and associations between data objects from a data set. In this way, interesting relations between objects in a data set can be found. This technique is often used in marketing, for example, which products are often bought together? If we translate this example to our problem, this technique may help us to identify which combinations of shifts are often combined.

The third type is *classification*, the process of finding a model that describes and distinguishes different classes. Based on characteristics of an object in the data set, the model provides a label to this object representing its class. The model is based on a training set, this set contains objects for which the class labels are known. In this research, we aim to find the scheduling preferences of customers. Since preferences are personal, each organization and department has unique preferences. Therefore, we do not have access to a training set including known preferences. For this reason, classification is no usable data technique in this thesis.

The fourth type is *cluster analysis*, which clusters objects in a data set based on their characteristics. In contrast to classification mining, clustering does not require a training set. In this research, cluster analysis might be useful to distinguish desirable patterns from undesirable patterns.

The fifth type is *outlier analysis*, where objects in a data set that do not comply with the general behavior are extracted. Outlier analysis is used in cases where rare events are interesting, for example in detecting fraudulent usage of credit cards. In this research, we are interested in frequent patterns in the data, as we assume that these frequent patterns represent the scheduling preferences. For this reason, we are not interested in the outliers. Therefore, outlier analysis is no usable data technique in this thesis.

Summarizing, in this research, *frequent patterns and association mining* and *cluster analysis* are usable data techniques to detect frequent patterns in data sets. Section 3.2.1 provides a description of frequent patterns and association mining and Section 3.2.2 provides a description of cluster analysis.

3.2.1 Frequent patterns and association mining

Frequent pattern and associations can be extracted from data sets by using *association rule mining*. Association rule mining is a data mining technique to extract interesting correlations, frequent patterns pr associations among sets of items in a database (Kotsiantis and Kanellopoulos, 2006). The input of association rule mining is a data set with multiple observations. In association rule mining one observation is called a transaction t, where $t \in T$. Each transaction consists of an item set i including one or multiple items, where $i \in I$. Table 3.1 provides an example of a data set including three transactions and four different items. In this example, Transaction t_1 includes the following item set: $\{i_1, i_3\}$.

In association rule mining, all possible item sets are created. The size of an item set is equal to n, which varies between 1 and the number of unique items in the data set, i.e. $I = \{i_1, i_2, ..., i_n\}$. The first step in association rule mining is to extract frequent item sets from the database. In

Table 3.1: Example of a data set including three transactions and four items

	i_1	i_2	i_3	i_4
t_1	1	0	1	0
t_2	0	1	1	0
t_3	1	1	1	1

this step item sets that occur infrequently are removed. Subsequently, association measures are calculated for each of the frequent item sets i. Finally, if the measures for an item set i satisfy a certain threshold, an association rule is created. In the remaining of this section, we explain these three steps more detailed.

1 **Frequent item set generation**

In general, the number of possible item sets i in a data set grows exponentially with the number of different items the data set contains: a data set that contains n items can generate up to 2^n–1 item sets (Tan et al., 2005). Consequently, the search space for frequent item sets that need to be explored grows exponentially. In the literature, various ways to efficiently select frequent item sets have been explored. A commonly used approach is the Apriori principle, which is based on the following principle: 'If an item set is frequent, then all of its subsets must also be frequent' (Tan et al., 2005). For example, if item set $\{i_2, i_3\}$ is frequent, then item set $\{i_2\}$ must also be frequent.

2 **Association measures**

Two important measures for association rules are support and confidence (Kotsiantis and Kanellopoulos, 2006). The support of an item set gives an idea of the frequency of that item set compared to all observations N in the data set, see Equation 3.1. Association rules indicate an implication between items in a data set. For example, if a transaction contains i_1 there is a high probability that this transaction also contains i_3. The rule of implication is formulated as follows: $X \rightarrow Y$, where X and Y represent an item set. For each rule of implication, association measures can be calculated that can be used to eliminate irrelevant association rules.

Equation 3.2 shows the formula for the *support* of an item set. Low *support* of an association rule indicates that the event occurred by chance. Therefore, the support reflects the usefulness of an association rule (Han et al., 2012b). Equation 3.3 shows the formula for the *confidence* of an item set. The *confidence* reflects the certainty of an association rule. The *confidence* equals the conditional probability $P(Y|X)$ (given that a transaction contains all items from item set X, the probability that the same transaction also contains all items from item set Y). The confidence can be high, even though there is no association between the items. Lift is a measure that can be used to overcome this problem, it compares the confidence with the expected confidence, see Equation 3.4. A lift smaller than 1 indicates a negative correlation between the items, a lift greater than 1 indicates a positive correlation and a lift equal to 1 indicates that the items are independent of each other. Another commonly used measure is conviction, see Equation 3.5. Conviction is a measure of implication, and not just co-occurrence, i.e. $conviction(X \rightarrow Y) \neq conviction(Y \rightarrow X)$ (Kotsiantis and

Kanellopoulos, 2006). In cases where the associations are directed, conviction is preferred over lift (Lin and Tseng, 2006).

Besides creating positive association rules, negative association rules can be created. Negative association rules indicate which items conflict with each other (Wu et al., 2004). Equations 3.6 up to 3.10 represent the association measures for negative association mining. For example, $support(X \rightarrow \neg\, Y)$, represents the support for transactions that include all items from item set X and does not include all items from item set Y.

Positive association mining measures

$$support(X) = \frac{frequency(X)}{N} \tag{3.1}$$

$$support(X \rightarrow Y) = \frac{frequency(X \rightarrow Y)}{N} \tag{3.2}$$

$$confidence(X \rightarrow Y) = \frac{frequency(X \rightarrow Y)}{frequency(X)} \tag{3.3}$$

$$lift(X \rightarrow Y) = \frac{support(X \rightarrow Y)}{support(Y)} \tag{3.4}$$

$$conviction(X \rightarrow Y) = \frac{1 - support(Y)}{1 - confidence(X \rightarrow Y)} \tag{3.5}$$

Negative association mining measures

$$support(\neg X) = 1 - support(X) \tag{3.6}$$

$$support(X \rightarrow \neg Y) = \frac{frequency(X \rightarrow \neg Y)}{N} \tag{3.7}$$

$$confidence(X \rightarrow \neg Y) = \frac{frequency(X \rightarrow \neg Y)}{frequency(X)} \tag{3.8}$$

$$lift(X \rightarrow \neg Y) = \frac{support(X \rightarrow \neg Y)}{support(\neg Y)} \tag{3.9}$$

$$conviction(X \rightarrow \neg Y) = \frac{1 - support(\neg Y)}{1 - confidence(X \rightarrow \neg Y)} \tag{3.10}$$

3 **Rule selection**

An association rule is called strong if it satisfies both a minimum support threshold and a minimum confidence threshold determined by the user (Han et al., 2012c). An unsuitable support threshold leads to the following problems: i) In the case of a too high support threshold, interesting patterns might be left out and ii) In the case of a too low support threshold, irrelevant patterns may be identified (Lin and Tseng, 2006). In different data sets, a different optimal minimum support threshold exists. In this research, we explore data

sets of different customers. Therefore, it is not possible to set a general minimum support threshold for all databases. Lin and Tseng (2006) propose a method to seek association rules with high confidence and a positive lift, without the need of a user-specified support threshold. The following three definitions are used to explain this method.

(a) **Automated support threshold**
An item set $I = \{i_1, i_2, ..., i_n\}$ is a frequent item set if the support of the item set I is larger than or equal to the lowest value of the support of the items in I, i.e,

$$support(I) \geq \min_{i \in I} support(i) \tag{3.11}$$

(b) **Strong association rules**
Furthermore, Lin and Tseng (2006) define the *minimum support* of an item i: $ms(i)$. An association rule $X \rightarrow Y$ is to be considered strong if the support of the association rule is larger than the lowest value of the minimum support of the items in the item set I, see Equation 3.12.

$$support(X \rightarrow Y) \geq \min_{i \in X \cup Y} ms(i) \tag{3.12}$$

The support of the items included in the item set, $support(i)$, are ordered in ascending order. The minimum support of the items is calculated as follows:

$$ms(i) = \begin{cases} support(i) * support(i+1) & if \quad 1 \leq i \leq n-1 \\ support(i) & if \quad i = n \end{cases} \tag{3.13}$$

Additionally, to be considered strong, the confidence of an association rule must exceed a user-specified minimum confidence threshold, see Equation 3.14. Lin and Tseng (2006) use a minimum confidence threshold of 0.5.

$$confidence(X \rightarrow Y) \geq MinimumConfidence \tag{3.14}$$

(c) **Interesting association rules**
A strong association rule is also considered as interesting if the lift is greater than 1, see Equation 3.15. As $lift(X \rightarrow Y) \geq 1$ only holds if $conviction(X \rightarrow Y) \geq 1$, lift and conviction are replaceable in this definition. Therefore, although the direction of association is relevant in this thesis, using lift as a measure in this context is correct.

$$lift(X \rightarrow Y) \geq 1 \tag{3.15}$$

3.2.2 Cluster Analysis

Cluster analysis is a data mining technique where a set of data objects is partitioned into different subsets. Objects are assigned to a subset in such a way that the objects within a subset are similar to another and dissimilar to objects in other subsets (Hair and Black, 2000). Each subset is called a cluster. Clustering is an unsupervised learning method, as the class label information

is not known on forehand. A frequently used partitioning method is K-Means clustering. The remaining of this section describes K-Means clustering based on *Cluster Analysis* from Hair and Black (2000).

Within K-Means clustering, the objects, o, in a data set are represented in a Euclidean space. The K-Means algorithm clusters these objects into k clusters. The objective function of the algorithm used in K-Means clustering aims for high similarity within a cluster and low similarity between clusters. In other words, the objective is to minimize the total-within-cluster variation and maximize the between-cluster variation. Optimization of the total-within-cluster variation, $SS^{TotalWithinCluster}$, is proven to be an NP-hard problem Hair and Black (2000). Therefore, in practice, a heuristic is often used to find a solution within reasonable time. The following steps are taken in K-Means clustering:

1 **Scale all objects to an Euclidean space**

2 **Specify the number of clusters to create, K**

3 **Arbitrarily select K objects**
The randomly selected objects represent the initial centers of the clusters, C_k.

4 **Assign each observation to the closest center** C_k
Subsequently, each object, o, in the data set is assigned to the closest center using $distance(o, c_k)$, the Euclidean distance between these points. The within-cluster-Sum of Squares (SS), $SS_k^{WithinCluster}$, represents the quality of a cluster k and can be calculated as follows:

$$SS_k^{WithinCluster} = \sum_{o \in C_k} distance(o, c_k)^2 \tag{3.16}$$

The total-within-cluster-SS equals:

$$SS^{TotalWithinCluster} = \sum_{k=1}^{K} SS_k^{WithinCluster} \tag{3.17}$$

5 **Update the cluster means**
For each cluster k a new cluster center is calculated given the objects assigned to this cluster. Subsequently, all the objects are reassigned using the new cluster centers.

6 **Continue until no improvement is found, or the maximum number of iterations has been reached**

Additionally, The $S^{BetweenClusters}$ indicates to what extent the clusters differ from each other. The $S^{BetweenClusters}$ is equal to the sum of squared distances between the centers, see Equation 3.18. K-Means clustering aims for a low $SS^{TotalWithinCluster}$ and a high $SS^{BetweenClusters}$.

$$SS^{BetweenClusters} = \sum_{k=1}^{K} \sum_{j=1}^{K} distance(c_k, c_j) \tag{3.18}$$

Using the K-Means algorithm brings the following *advantages*:

- Simple and fast algorithm
- Can be applied effectively on small- to medium-size data sets

and the following *disadvantages*:

- If the number of clusters K is unknown, it is difficult to determine number of clusters K
  A method to overcome this problem is the elbow method. The elbow method is based on the following observation: increasing the number of clusters, reduces the number of objects assigned to each cluster and therefore can reduce the within-cluster variation of each cluster. However, the marginal reduction of the within-cluster variation may decrease if the number of clusters increases. In other words, at some point increasing the number of clusters does not have much effect on the solution. Consequently, the curve of the total within-cluster variation and the number of clusters formed shows a so-called 'elbow'. The number of clusters K is chosen as the turning point in this curve.
- The results depend on the initially randomly chosen centers
  A drawback of the K-Means algorithm is that it does not have to converge to a global optimum and can terminate at a local optimum. Consequently, the results may be dependent on the initial random centers. Therefore, it is recommended to apply the K-Means multiple times with different initial centers and select the best-found solution.
- Sensitive to outliers
  The K-Means method is sensitive to outliers. Additionally, changing the order of the data may result in different clusters.

3.3 Evaluation of results

This section describes evaluation metrics to evaluate the performance of data mining techniques. This section is divided into two sub-sections. Section 3.3.1 describes evaluation metrics to evaluate nominal variables. Nominal variables include mutually exclusive categories. For nominal values, the order between the categories does not matter. Section 3.3.2 describes evaluation metrics to evaluate ratio variables. Ratio variables are non-categorical variables. For ratio variables, the order matters and the difference between two values is meaningful.

3.3.1 Nominal variables

As described before, nominal variables include mutually exclusive categories. Each of these categories is called a class. In classification problems, data mining techniques are used to assign a variable to one of these classes. Such a data mining technique is a so-called classifier. A confusion matrix, see Figure 3.1, helps to evaluate the performance of such a classifier. The example in Figure 3.1 shows a confusion matrix for a nominal variable including two classes, in other words, a binary variable. A confusion matrix can also be created for nominal variables including multiple classes.

We explain the confusion matrix using an example. An instance can take the value *true* or *false*. A given data set includes a number of instances, where the actual value (*true* or *false*)

	Identified class			
	Classes	*True*	*False*	Total
Actual class	*True*	TP	FN	P
	False	FP	TN	N
	Total	P'	N'	P + N

Figure 3.1: Confusion matrix of a classification problem with two classes

is known. In the confusion matrix, the P and N represent the number of instances that are equal to *true* and *false*, respectively. In a classification problem, the objective is to identify the value of an instance, without knowing its actual value. A classifier identifies the value for every instance. In the confusion matrix, the P' and N' represent the number of instances classified as *true* or *false*, respectively. To determine the performance of a classifier, the actual and classified value of an instance is compared. For example, if the actual value of an instance is *true* and the instance is classified as *false*, the instance is labeled as false negative. Table 3.2 explains all definitions used in the confusion matrix. The confusion matrix can be used to calculate measures to evaluate the performance of a classifier method, in the remaining of this section we give a short description of these measures based on Han et al. (2012c).

Table 3.2: Definitions evaluation of classification method

True Positive (TP)	The number of instances classified as true that are actual true
True Negative (TN)	The number of instances classified as false that are actual false
False Negative (FN)	The number of instances classified as false that are actual true
False Positive (FP)	The number of instances classified as true that are actual false
P'	The number of instances classified as true
N'	The number of instances classified as false
P	The number of instances that actual true
N	The number of instances that actual false

The accuracy of the method is the percentage of variables that have been classified correctly, whereas, the error rate is the percentage of variables that have been classified incorrectly, see Equations 3.19 and 3.20. Accuracy and error rates give equal weight to false positives and false negatives, therefore, positive and negative instances should be of equal importance.

Other common methods to evaluate the model are recall and specificity, see Equations 3.21 and 3.22. The recall, also known as the true positive rate, is equal to the percentage of positives instances that have been classified positive. Recall is also known as a measure of completeness. In cases where it is important not to miss any true instances a high recall is required. The specificity, also known as the false positive rate, is equal to the percentage of false instances that have been classified false. Cases where it is important not to classify false instances as true require a high specificity. In general, methods with high recall, have a lower specificity.

Precision is a measure of exactness, meaning, the percentage of instances that are classified as true is actually true, see Equation 3.23. Finally, an F_1-Measure combines the measures of precision and recall in one single-measures, see Equation 3.24. The F_1-Measure assigns equal

weight to precision and recall, whereas the F_β gives β times more weight to recall as to precision. The aforementioned measures can be calculated as follows:

$$Accuracy = \frac{TP + TN}{P + N} \tag{3.19}$$

$$Error = 1 - Accuracy \tag{3.20}$$

$$Recall = \frac{TP}{P} \tag{3.21}$$

$$Specificity = \frac{TN}{N} \tag{3.22}$$

$$Precision = \frac{TP}{TP + FP} \tag{3.23}$$

$$F_\beta = \frac{(1 + \beta)^2 * precision * recall}{\beta^2 \ * precision \ + \ recall} \tag{3.24}$$

3.3.2 Ratio variables

This section describes metrics to evaluate the identified values of ratio variables based on Drakos (2018). A frequently used measure to evaluate continuous variables is the Mean Squared Error (MSE), see Equation 3.25. The MSE measures the average squared error of the identified values compared to the actual values. The Root Mean Squared Error (RMSE) is introduced to scale the errors to the same size of the target values and is equal to the root of the MSE, see Equation 3.26. As the error in the MSE and the RMSE is squared, both methods penalize bigger errors more than smaller errors. The Mean Absolute Error (MAE) takes the absolute values of the errors without squaring them, see Equation 3.27. Therefore, the MAE is less sensitive to outliers compared to RMSE and MSE.

Based on MSE, RMSE or MAE it is difficult to determine to what extent the method can identify the right values. The coefficient of determination, R^2, compares the performance of the model and a baseline model, see Equation 3.28. The baseline model is the simplest possible model, for example, setting the identified value equal to the average of all items. The value of R^2 lies between $-\infty$ and 1, a negative value indicates that the model is worse than the baseline model, a value close to 0 indicates that the performance of the model is close to the performance of the baseline model and a value close to 1 means that the model has almost zero error.

$$MSE = \frac{1}{N} \sum_{i=1}^{N} (y_i - \ \hat{y}_i)^2 \tag{3.25}$$

$$RMSE = \sqrt{MSE} \tag{3.26}$$

$$MAE = \frac{1}{N} \sum_{i=1}^{N} |y_i - \ \hat{y}_i| \tag{3.27}$$

$$R^2 = 1 - \frac{MSE(model)}{MSE(baseline)} \tag{3.28}$$

3.4 Conclusion

The objective of this chapter is to answer the following research question: *What can we learn from the literature about identifying preferences from data?* The following sub-questions have been answered in this chapter:

a *What techniques exist to learn soft constraints from historical data?*
Relatively few approaches exist to learn soft constraints from historical data. In most approaches, user-feedback is required or only the weights of the constraints are being learned. As this research aims to identify the parameters of soft constraints without using user-feedback, these existing approaches cannot be applied in this research. Additionally, the described approaches tackle small theoretical problems rather than complex real-world problems.

As suggested by Campigotto et al. (2011), known hard constraints can be used to reduce the solution space of complex problems, and thus the number of possible solutions. Using hard constraints, therefore, helps to structure complex problems and can be used to determine the boundaries of the soft constraints. This allows for a more targeted search for soft constraints within the data.

b *What data mining techniques can be used to identify preferences from data?*
Data mining is the process of identifying interesting patterns in data sets. The literature describes several categories of interesting patterns. Two of these patterns are *frequent patterns and associations mining* and *cluster analysis*. First, association rule mining can be applied to identify frequent and infrequent item sets from data sets. In Chapter 1, the assumption was made that historical shift schedules contain implicit preferences. We assume that desirable scheduling decisions were made frequently. Mining frequent patterns helps us to identify these frequently made scheduling decisions. Furthermore, infrequent patterns could indicate undesirable scheduling decisions. Second, K-Means clustering is a data mining technique to cluster objects in a data set based on their characteristics. This technique could help us to cluster employees within a scheduling department, as different preferences may exist for part-time and full-time employees.

c *What evaluation metrics are available to determine the performance of a data mining technique?*
Evaluation metrics are a valuable tool to evaluate the performance of a data mining technique. In the literature, a distinction is made between the evaluation of ratio and nominal variables. First, a confusion matrix is a useful evaluation tool for nominal parameters. It represents the number of correct and incorrect classified items. Using a confusion matrix, several measures can be calculated to quantify of the results. Second, a frequently used measure to evaluate ratio parameters is the MSE. Several variations on the MSE have been proposed in the literature.

Chapter 4

Problem approach

Chapter 2 describes types of soft constraints used by ORTEC's customers who currently use the Optimizer. In this chapter, we propose a method to identify the parameters of these soft constraints. As described in Chapter 1, the objective is to use both historical shift schedules and domain knowledge to identify scheduling preferences of ORTEC's customers. With domain knowledge, we mean specific knowledge about shift scheduling in practice, for example, labor laws and known preferences. Adding domain knowledge to the data mining technique prevents us from suggesting irrelevant constraints to the customer and could improve the performance of the method we propose. Figure 4.1 shows a schematic overview of the problem approach.

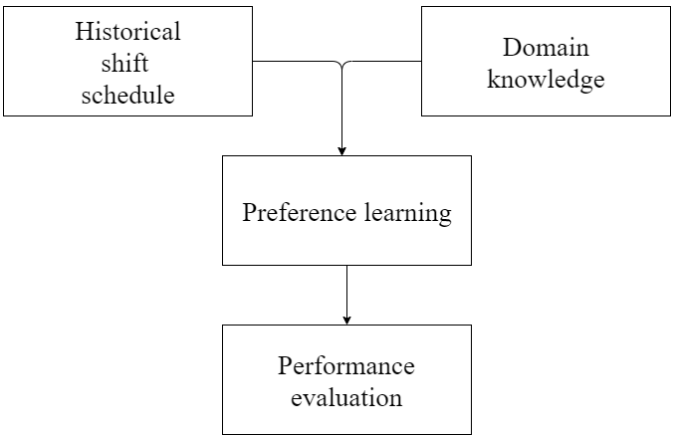


Figure 4.1: Schematic overview of the problem approach

Each section in this chapter is devoted to one of the steps shown in Figure 4.1. Section 4.1 defines relevant constraints, the parameters to identify and the data to extract from the historical schedule. Section 4.2 describes how valuable domain knowledge can be used to define the boundaries of the soft constraints. Section 4.3 describes data mining techniques to learn the parameters of the constraints defined. Section 4.4 describes evaluation metrics to evaluate the performance of the method we propose. Finally, the chapter ends with a conclusion in Section 4.5 to answer the following research question: *How can we identify the parameters of relevant soft constraints?*

4.1 Constraint formulation

As described in Chapter 2, the customers included in this research use seven types of soft constraints. To structure the problem, we assign these constraints to the following four different categories: i) Consecutive duties, ii) Free weekends, iii) Series length and iv) Distribution of night shifts. Table 4.1 shows for each soft constraint to which category it belongs. In this section, we formulate these preferences mathematically. For each constraint, we define one or more parameter(s) C, representing the parameters we aim to identify based on historical schedules. To identify preferences based on historical data, we assume that preferred patterns occur more frequently than non-preferred patterns. To measure the frequency of patterns in historical data, we define for each constraint one or more parameter(s) x. This section includes four subsections, each subsection is devoted to one of the categories described in the table.

Table 4.1: Categories of soft constraints

Category	**Soft constraint**
Consecutive duties	Avoid combinations
	Change in begin time
	Same duty
Free weekends	Number of free weekends
Series length	Preferred series length
Distribution of night shifts	Number of night shifts
	Preferred series length night shifts

4.1.1 Consecutive duties

We define consecutive shifts as shifts that are worked on two succeeding days by the same employee. Next, we define consecutive duties as the duties combined in two consecutive shifts. The soft constraints *avoid combinations*, *change in begin time* and *same duty* indicate to what extent it is desirable to schedule shifts with specific duties on two succeeding days. We combine these three soft constraints in one constraint type: *Consecutive duties*.

We define $C_{a,b}^{combi}$ to indicate the desirability to assign shifts with duty type a and b to the same employee on two succeeding days. If the planners finds it desirable to schedule shifts with duty a and duty b on succeeding days then $C_{a,b}^{combi}$ takes a negative value (-1). A positive value (1) of $C_{a,b}^{combi}$ indicates that it is undesirable to schedule shifts with duty a and b on succeeding days. A neutral relation between two duty types can be indicated to set $C_{a,b}^{combi}$ equal to 0. Next, we define $x_{a,b}^{combi}$ to measure the frequency that a shift with duty a was followed by a shift with duty b in the historical shift schedule.

Equation 4.1 shows how the severity is calculated for violating this constraint. As described in Chapter 2, violating a soft constraint incurs a penalty for the objective function of the minimization problem. In our formulation of this constraint type, the severity can take a negative value. In this way we introduce a bonus for the objective value, i.e. adding up a minus *severity* to the objective function.

$$Severity^{combi} = \sum_{a \in D} \sum_{b \in D} C_{a,b}^{combi} * x_{a,b}^{combi} \tag{4.1}$$

Where:

$$x_{a,b}^{combi} = \text{the frequency that a shift with duty } a \text{ is followed by a shift with duty} \tag{4.2}$$

$$b \text{ on a succeeding day} \tag{4.3}$$

$$C_{a,b}^{combi} = \begin{cases} -1, & \text{desirable to schedule shifts with duty } a \text{ and } b \text{ on succeeding days} \\ 1, & \text{undesirable to schedule shifts with duty } a \text{ and } b \text{ on succeeding days} \\ 0, & \text{otherwise} \end{cases} \tag{4.4}$$

4.1.2 Free weekends

The constraint *Number of free weekends* stands on its own, therefore one category is devoted to this constraint: *Free Weekends*. In OWS the constraint is formulated as follows: *In a period of [NWeeks], an employee should have at least [NWeekends] weekends off.* An employee has a free weekend if no work takes place between Saturday 04:00 and Sunday 23:59.

We define $C_p^{weekends}$ to indicate the preferred minimum number of free weekends in a period of p weeks. Additionally, we define $x_{n,p}^{weekends}$ to measure the frequency of free weekends in historical data, where n is the number of free weekends and p the length of the corresponding period in weeks. We measure the number of free weekends by using a rolling horizon with a time window of p weeks. Figure 4.2 shows an example of how the number of free weekends is measured for a p equal to 4 weeks.

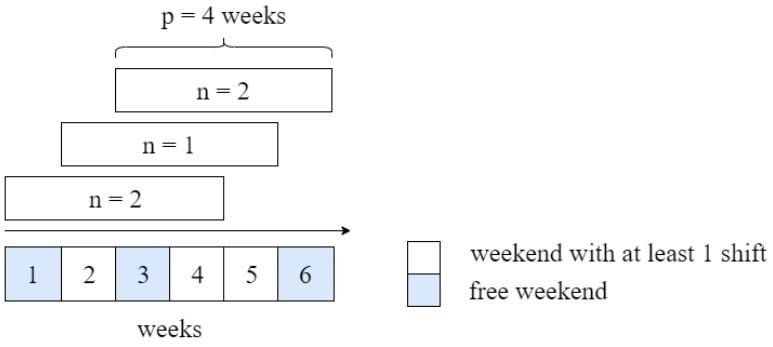


Figure 4.2: Example of measurement of the number of free weekends for a period length of 4 weeks

In OWS, this constraint type has quadratic costs. This means that the deviation is squared. In this way, larger deviations from the preference are considered to be more severe. For this reason, in this thesis, we also use a quadratic cost type for this soft constraint. Moreover, this constraint defines the *minimum* of free weekends an employee should have, therefore, a penalty is assigned only if an employee has too few free weekends in a given period. Equation 4.5 shows the calculation for the severity of violating the constraint *free weekends*. Equation 4.8 determines the boundaries for the soft constraint. To fulfill demand, it is not reasonable to wish that employees are all weekends free, therefore, the number of preferred free weekends can be at most $p - 1$.

$$Severity^{weekends} = \sum_{p \in P} \sum_{n=1}^{p-1} x_{n,p}^{weekends} * \left(max \left\{ 0, \quad C_p^{weekends} - \ n \right\} \right)^2 \tag{4.5}$$

Where:

$$x_{n,p}^{weekends} = \text{frequency of } n \text{ weekends free in a period of } p \text{ weeks} \tag{4.6}$$

$$C_p^{weekends} = \text{the preferred minimum number of free weekends in } p \text{ weeks} \tag{4.7}$$

$$0 \leq C_p^{weekends} \leq p - 1 \quad \forall \quad p \tag{4.8}$$

4.1.3 Series length

The third category includes the following soft constraint: *preferred series length*. In OWS, this soft constraint is formulated as follows: *A series of shifts should have a minimum length of [MinLength] and [MaxLength]. This constraint holds only for employees with a contract of minimal [MinContractHours] and maximal [MaxContractHours].* This constraint has two different types of parameters: i) The first set of parameters, *MinLength* and *MaxLength*, define the boundaries of the preferred series length and ii) The second set of parameters, *MinContractHours* and *MaxContractHours*, define the group of employees this constraint holds for.

We define G a set of employee groups g. Employees are assigned to an employee group based on their contract hours. Later in this chapter, we describe how these groups are defined. To indicate the preferred length of a series for each employee group, we define $C_g^{MinLength}$ and $C_g^{MaxLength}$. To measure the frequency that series lengths occur in the historical schedule, we define $x_{g,l}^{SeriesLength}$, the number of shifts that belong to a series of length l per employee group g. Figure 4.3 shows how the lengths of a series length are measured in an example shift schedule.

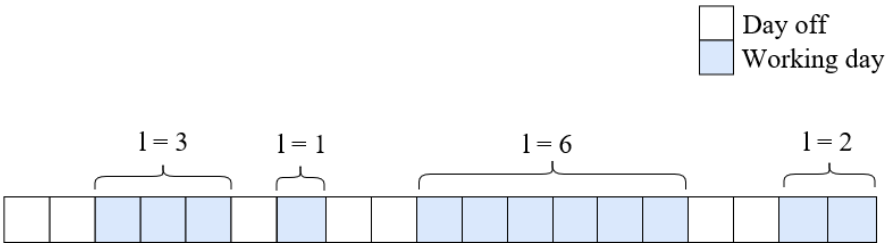


Figure 4.3: Example measurement of series lengths

A penalty is added to the objective function if an employee works a series of a length outside the preferred boundaries. As larger deviations from the preference are considered to be more severe, this constraint type has a quadratic cost type in OWS. Equation 4.9 shows how the severity of violating this constraint is calculated. Additionally, we define Equation 4.14, to make sure that the identified maximum length is at least as big as the identified minimum length.

$$Severity^{Series} = \sum_{g \in G} \sum_{l \in L} x_{g,l}^{SeriesLength} * \left(\max \left\{0, \ C_g^{MinLength} - l \ , \ l - \ C_g^{MaxLength} \ \right\}\right)^2 \tag{4.9}$$

Where:

$$C_g^{MinLength} = \text{preferred minimum length of a series for employee group } g \tag{4.10}$$
$$C_g^{MaxLength} = \text{preferred maximum length of a series for employee group } g \tag{4.11}$$
$$x_{g,l}^{SeriesLength} = \text{number of shifts belonging to a series of length } l \text{ worked by} \tag{4.12}$$
$$\text{employee group } g \tag{4.13}$$
$$C_g^{MinLength} \leq C_g^{MaxLength} \tag{4.14}$$

4.1.4 Distribution of night shifts

The final category defines how night shifts should be distributed among employees. This category contains the following two constraint types: *Preferred series length night shifts* and *Number of night shifts*.

Preferred series length night shifts

The constraint *Preferred series length of night shifts* defines the preferred minimum and maximum series length including night shifts. The series is interrupted if the employee did not work a night shift between 00:00 and 23:59. We define $C^{MinSeriesNight}$ and $C^{MaxSeriesNight}$ to indicate the preferred minimum and maximum series length for night shift series, respectively. Additionally, we define $x_l^{SeriesNight}$, the number of night shifts that belong to a series of length l. These parameters are similar to the parameters of the constraint *Series length*. However, in contrast to preferences for series lengths, preferences for the series length of night shifts hold for all employees regardless of their contract hours. Therefore, we do not define employee groups for this constraint type. Equation 4.15 shows the calculation for the severity of violating this constraint.

$$Severity^{SeriesNight} = \sum_{l \in L} x_l^{SeriesNight} * \left(\max \left\{0, \ C^{MinSeriesNight} - l \ , \ l - \ C^{MaxSeriesNight} \ \right\}\right)^2 \tag{4.15}$$

Where:

$$C^{MinSeriesNight} = \text{preferred minimum length of a series of night shifts} \tag{4.16}$$
$$C^{MaxSeriesNight} = \text{preferred maximum length of a series of night shifts} \tag{4.17}$$
$$x_l^{SeriesNight} = \text{number of night shifts belonging to a series of length } l \tag{4.18}$$
$$C^{MinSeriesNight} \leq C^{MaxSeriesNight} \tag{4.19}$$

Number of night shifts

In OWS, the constraint is defined as follows: *Employees should work between [Minimum] and [Maximum] number of night shifts in a period of [NWeeks] weeks.* We define $C_p^{MinNightShifts}$

and $C_p^{MaxNightShifts}$ as the preferred minimum and maximum number of night shifts an employee should work in p weeks, respectively. Additionally, we define $x_{n,p}^{NightShifts}$ to measure the frequency that employees worked n night shifts in p weeks. A penalty is added to the objective function if an employee works too less or too many night shifts in a period. Equation 4.20 shows the calculation for the severity of violating the constraint number of night shifts. Equation 4.26 indicates that the preference for the maximum number of night shifts should be at least as large as the minimum number of night shifts.

$$Severity^{NightShifts} = \sum_{p \in P} \sum_{n \in N} x_{n,p}^{Frequency} * (max\{0, C_p^{MinNightShifts} - n, n - C_p^{MaxNightShifts}\})^2 \tag{4.20}$$

Where:

$$C_p^{MinNightShifts} = \text{minimum number of night shifts an employee should work} \tag{4.21}$$

$$\text{in } p \text{ weeks} \tag{4.22}$$

$$C_p^{MaxNightShifts} = \text{maximum number of night shifts an employee should work} \tag{4.23}$$

$$\text{in } p \text{ weeks} \tag{4.24}$$

$$x_{n,p}^{NightShifts} = \text{frequency that employees work } n \text{ night shifts in } p \text{ weeks} \tag{4.25}$$

$$C_p^{MinNightShifts} \leq C_p^{MaxNightShifts} \tag{4.26}$$

4.2 Domain knowledge

ORTEC has much domain knowledge about shift scheduling and scheduling preferences from customers from years of experiences. This section describes the available domain knowledge for each of the categories of soft constraints. This section is divided into four subsections. Each subsection is devoted to one of the categories of soft constraints.

4.2.1 Consecutive duties

The objective is to identify $C_{a,b}^{combi}$ for every combination of duties, which indicates the desirability of working duty a and duty b on succeeding days. To prevent us from suggesting irrelevant preferences to the customers, it is important to know if a combination of duties did not occur in history because it is undesirable or it is not possible due to hard constraints. In this way, hard constraints define the boundaries of soft constraints. We define $ForbiddenDuties_{a,b}$, which indicates if duty type a and b cannot be performed on two succeeding days by the same employee.

$$ForbiddenDuties_{a,b} = \begin{cases} 0 & \text{duty } a \text{ can be followed by duty } b \text{ on succeeding days} \\ 1 & \text{otherwise} \end{cases} \tag{4.27}$$

The following hard constraints determine the value of $ForbiddenDuties_{a,b}$:

a **Required skills**

In most organization, a duty contains a required skill. This means that only employees

owning that skill, are allowed to perform this duty. In this way, skills reduce the number of duty types an employee can work. We explain this using an example. Within a nursing department, a receptionist cannot work a doctor's shift and the other way around. Consequently, we know that this combination of duties, receptionists duty and doctors duty, is forbidden by hard constraints. Therefore, if none of the employees is qualified to work both duties a and b, the combinations of duties is forbidden.

b **Duty demand**

To use the optimizer, departments set a duty demand; Table 4.2 shows an example of such a duty demand table. Since there is no demand for duty 1 and duty 3 on two succeeding days, it is not possible to schedule these duties on succeeding days. In this way, the duty demand table helps us to remove irrelevant combinations of consecutive duties.

Table 4.2: Example of a duty demand table

Weekday	**Monday**	**Tuesday**	**Wednesday**	**Thursday**	**Friday**	**Saturday**	**Sunday**
Duty 1	0	0	0	0	0	5	5
Duty 2	2	2	2	2	2	2	2
Duty 3	0	3	4	0	0	0	0

c **Labor laws**

Labor laws regulate the maximum number of hours an employee may work per day and the minimum number of hours rest required after a day of work. If scheduling two duties on succeeding days results in a violation of one of these labor laws, the combination of duties is forbidden. The customer cases in this research are located in The Netherlands or Finland, therefore, we describe the relevant regulations in these countries for combining duties on two succeeding days.

The Netherlands:

The following rules defined defined by Ministerie van Sociale Zaken en Werkgelegenheid (2010) affect the allowed combinations of consecutive duties:

- 'After a working day, an employee must have 11 consecutive hours of non-work time. This rest period may be shortened to 8 hours once in a 7-day period if the nature of the work or the business circumstances requires this.'
- 'An employee works a night shift if he must work for more than 1 hour between the hours of midnight and 6 am. If a night shift ends after 2 am, this must be followed by a minimum of 14 hours of non-work time. This may be shortened to 8 hours a maximum of once per week. But only if the type of work or the business circumstances necessitates this.'
- 'No on-call duty 11 hours before or 14 hours after a night shift.'

Finland:

For the Finnish customers, the following hard constraint regulates affects the allowed combinations of consecutive duties(Ministry of Economic Affairs and Employment of Finland, 2017):

- 'An employee is entitled to an uninterrupted rest time of at least 11 hours within 24 hours from the beginning of each work shift.'

Summarizing, the following hard constraints are used to determine forbidden combinations of duties: i) Required skills, ii) Duty demand and iii) Labor laws. By defining forbidden combinations of consecutive duties, we remove irrelevant soft constraints.

4.2.2 Free weekends

Organizations may have a preference for the minimum number of free weekends employees should have. To what extent a planner can adhere to this preference, strongly depends on the demand for weekend shifts in a department. For example, if an employee is required to work 6 weekend shifts a month, it is not reasonable to prefer that every employee has at least 1 out of 2 weekends off. Theoretically, the number of weekend shifts an employee should work each week can be calculated using the following formula:

$$\text{Number of weekend shifts per week} = \frac{\text{Demand weekend shifts per week}}{\text{Total number of employees}} \tag{4.28}$$

However, in practice, we observe that the number of weekend shifts is not equally distributed among the employees. The way weekend shifts are distributed depends on several external factors; such as the skills required to work weekend shifts, the preferences of the employees, the number of contract hours and other underlying hard and soft constraints. Figure 4.4 shows the distribution for the average number of weekends worked per employee for customer cases B1, B2 and B3. This figure shows that the number of weekend shifts worked per week varies per employee. Consequently, it is not possible to use Equation 4.28 to determine the number of weekend shifts an employee is required to work.

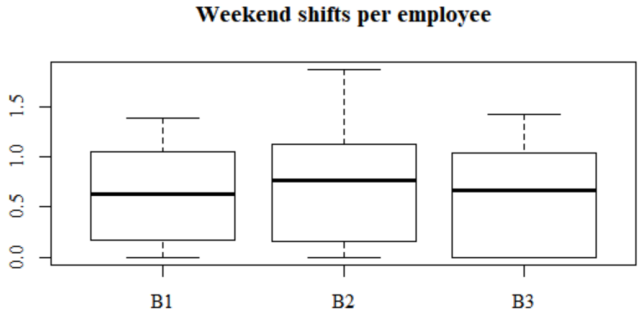


Figure 4.4: Average number of weekend shifts worked per employee for customer case B1, B2 and B3 measured in a scheduling period of 200 weeks.

The following steps make sure that only the relevant data is included in the analysis:

- *Selection of period*
  Figure 4.5 shows a plot of the weekly number of weekend shifts worked over time for customer B1. This figure shows that the number of weekend shifts worked has dropped over time. From week 50, the number of weekends shifts worked stabilizes. Since the demand for weekend shifts affects the number of free weekends per employee, we select

the period where the demand for weekend shifts is similar to the current demand. In this example, we do not consider the historical shift schedule before week 50.

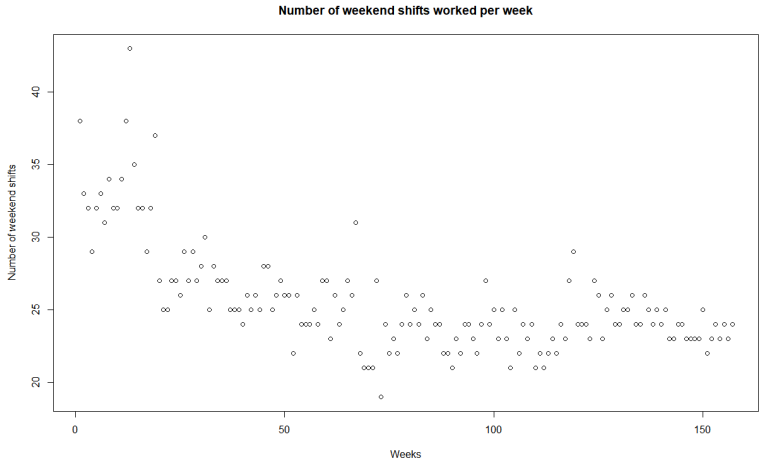


Figure 4.5: Total number of weekends shifts worked every week in department B1. In this example, the historical shift schedule before week 50 is removed from the analysis.

- *Remove week offs and sickness*
  Employees may not have worked during a given week due to, for example, sickness or vacation. If an employee did not work any shifts during a given week the observation is removed from the data. In this way, the number of free weekends is not affected by other underlying reasons.

- *Select employees with the highest burden*
  As explained before, the preference for the minimum number of free weekends depends on the number of weekend shifts employees are required to work. Among other reasons, organizations state scheduling preferences to make sure that employees get a healthy shift schedule. Therefore, we assume that the preference for the minimum number of free weekends holds for employees that experience the highest burden. For this reason, we only include shift schedules of employees working more than the median number of weekend shifts.

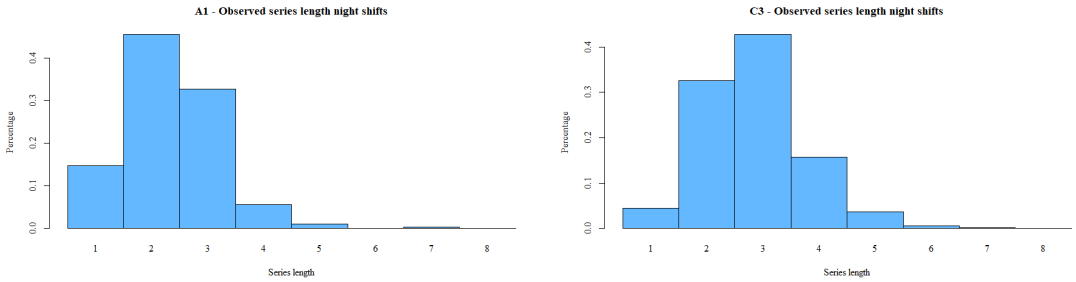
4.2.3 Series length

Both the Dutch and Finnish law do not limit the number of shifts an employee is allowed to work in a row. Only the minimum number of hours of uninterrupted rest per week is defined, however, this law does not necessarily affect this soft constraint.

Therefore, we use domain knowledge from consultants to define the boundary of this soft constraint. From practice, we know that the preferred series length does not exceed 8 shifts. Therefore, we consider a series length above 8 as outliers for all customers. In this way, we determine the upper bound for the soft constraint *series length*.

4.2.4 Distribution of night shifts

Working night shifts strongly influences the natural sleep rhythms of employees. Consequently, more strict rules apply for working night shifts (Ministerie van Sociale Zaken en Werkgelegenheid, 2010). To respect the natural sleep rhythm of employees as much as possible, organizations prefer to schedule multiple night shifts in a row, followed by a period of working day shifts. In this way, employees do not have to switch frequently between their day and night rhythm. This preference of the customers can be verified by having a look at the data, see Figure 4.6. This figure shows the observed distribution of the length of night shift series. We observe that in both cases single night shifts have a much lower frequency than night shifts worked in a series.



(a) Customer A1 (b) Customer C3

Figure 4.6: Observed distribution of night shift series for customer A1 and customer C3

Based on domain knowledge and observations in historical data, we assume that organizations prefer that employees work a series of night shifts over working single night shifts. Furthermore, we aim to identify the number of night shifts an employee should work in a given period. Therefore, we analyze the frequency that an employee works a series of night shifts. Figure 4.7 shows the pattern of the night shifts worked per week in the past two years, 2017 and 2018, by one employee. The blue bars show that an employee works 1 or 2 night shifts in a given week, thereafter, the employee is discharged from working night shifts for a couple of weeks.

Summarizing, the soft constraints *preferred series length night shifts* and *number of night shifts* are dependent on each other. First, the preferred series length of night shifts defines how long a series of shifts should be. Second, the frequency that such a series of night shifts occurs, defines the preferred number of night shifts in a given period.

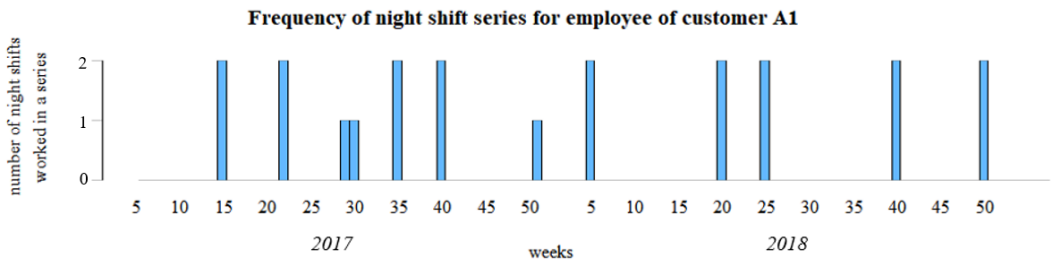


Figure 4.7: Department A1 - pattern of night shifts worked in each week in 2017 and 2018 for one employee. The blue bars represent the number of night shifts worked in a given week. The white spaces mean that the employee did not work a night shift in this week.

Furthermore, in an expert interview with an ORTEC consultant, we obtained the following domain knowledge:

- The personal circumstances of employees influences their preference for working night shifts (S. Hofstra, personal communication, November 15, 2019). One of the reasons for not willing to work night shifts is having young children. Whereas, other employees may prefer to work night shifts due to the extra allowances that come with this.
- According to the Dutch WHA, employees over 55 years are not *required* to work night shifts (Ministerie van Sociale Zaken en Werkgelegenheid, 2010). Therefore, we expect that an employee's age affects the number of night shifts the employee works. However, due to privacy reasons, no data is available about the age of employees.

Based on this information, we expect that employees have an individual preference for working night shifts. Figure 4.8 shows the number of night shifts worked per employee in a period of 6 months. This figure shows that the number of night shifts an employee works varies between individual employees. Currently, in OWS the constraint *number of night shifts* is defined on department level. Based on domain knowledge and historical data, we expect this is not a good representation of the reality to formulate the soft constraint on department level.

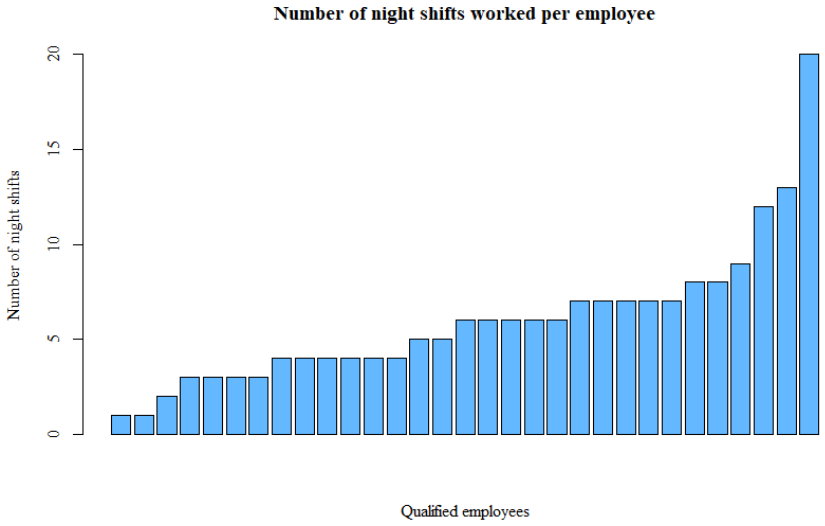


Figure 4.8: Customer A1: Number of night shifts worked per employee in a period of 6 months (from 01-04-2018 to 30-09-2019).

In conclusion, we recommend ORTEC to change the current formulation of the constraint *number of night shifts* as follows:

- Since employees preferably work night shifts in a series, the constraints should describe the frequency that an employee should work a series of night shifts rather than the number of night shifts in a given period.
- Since employees have personal preferences for working night shifts, the constraint should be formulated on group or individual level rather than on department level.

4.3 Preference learning

The objective is to identify the parameters, C, for each of the defined soft constraints. As described before, to achieve this, we measure the frequency of patterns in historical schedules and use domain knowledge to define the boundaries of the soft constraints. In this section, we propose a method for each of the soft constraints to identify its parameters. This section is divided into four subsections, each subsection is devoted to one of the categories of soft constraints.

4.3.1 Consecutive duties

As described in Chapter 3, association rule mining is a data technique that can extract frequent and infrequent patterns from data. Assuming that frequent patterns represent scheduling preferences and infrequent patterns represent aversions, association rule mining can be used to define soft constraints. Association rule mining creates implication rules, for example, $a \rightarrow b$ is an association rule that says the following: if an employee works duty a, then it is likely that the employee works duty b on the next day. This association rule indicates that a and b are *desirable* combinations of consecutive duties. The remainder of this section describes the steps to be taken to create these association rules and define desirable and undesirable combinations of consecutive duties.

1 **Create item sets**
The first step of association rule mining is to create item sets. Each item set represents a combination of consecutive duties. The item sets include succeeding duties, therefore, the order is important. As described in Chapter 3, the number of item sets grows exponentially with the number of items in a data set (Tan et al., 2005). To overcome this problem, item sets should be created efficiently, for example, using the Apriori principle described in Chapter 3. However, in this research, each ordered item sets includes exactly two items: duty a and b. Consequently, the number of possible item sets is at most k^2, where k equals the number of unique duties. Additionally, we use known hard constraints to create a forbidden duties matrix, which reduces the number of possible combinations of consecutive duties and thus reduces the number of item sets. Since the number of item sets is limited in this research, there is no need to efficiently create item sets. Summarizing, for every non-forbidden combination of duties a and b an ordered item set {a,b} is created.

2 **Calculate association measures**
Next, we calculate association measures to determine if an association rule should be created. First, we measure the frequency that a shift with duty a was followed by a shift with duty b on succeeding days, $x_{a,b}^{combi}$, in historical schedules. Second, The following association measures are calculated:

- $support(a)$: the relative frequency of events that start with a shift with duty a, see Equation 4.29.
- $support(b)$: the relative frequency of events that end with a shift with duty b, see Equation and 4.30.

- $support(\neg b)$: the relative frequency of events that were not followed by a shift with duty b, see Equation 4.31.
- $support(a \rightarrow b)$: the relative frequency of the event that a shift with duty a was followed by a shift with duty b on succeeding days, see Equation 4.32.
- $support(a \rightarrow \neg b)$: the relative frequency of the event that a shift with duty a was followed by shift with another duty than duty b on succeeding days, see Equation 4.33. This is a measure with negative implication, which gives information about undesirable combinations of duties.
- $confidence(a \rightarrow b)$: the probability that an employee who works a shift with duty a, works a shift with duty b the next day, i.e. the conditional probability $(a \mid b)$, see Equation 4.34.
- $confidence(a \rightarrow \neg b)$: the probability that an employee works a shift with duty a, does not work a shift with duty b the next day, i.e. the conditional probability $(a \mid \neg b)$ see Equation 4.35.
- $lift(a \rightarrow b)$: indicates if a correlation exists between the consecutive duties, see Equation 4.36. The confidence can be high, even though there is no correlation between the duties. The lift compares the confidence with the expected confidence to indicate the correlation. A lift greater than 1 indicates a positive correlation, and a lift smaller than 1 a negative correlation.

$$support(a) = \frac{\sum_{b \in D} x_{a,b}^{combi}}{N} \tag{4.29}$$

$$support(b) = \frac{\sum_{a \in D} x_{a,b}^{combi}}{N} \tag{4.30}$$

$$support(\neg b) = \frac{\sum_{a \in D} \sum_{b' \in D, b' \neq b} x_{a,b'}^{combi}}{N} \tag{4.31}$$

$$support(a \rightarrow b) = \frac{x_{a,b}^{combi}}{N} \tag{4.32}$$

$$support(a \rightarrow \neg b) = \frac{\sum_{b' \in D, b' \neq b} x_{a,b'}^{combi}}{N} \tag{4.33}$$

$$confidence(a \rightarrow b) = \frac{x_{a,b}^{combi}}{\sum_{b' \in D} x_{a,b'}^{combi}} \tag{4.34}$$

$$confidence(a \rightarrow \neg b) = \frac{\sum_{b' \in D, b' \neq b} x_{a,b'}^{combi}}{\sum_{b' \in D} x_{a,b'}^{combi}} \tag{4.35}$$

$$lift(a \rightarrow b) = \frac{1 - support(A \rightarrow B)}{1 - confidence(A \rightarrow B)} \tag{4.36}$$

where N is the total number of observations.

3 **Create association rules**

These association measures can be used to formulate association rules. We create both positive and negative rules.

- Positive rules: $a \rightarrow b$
  If an employee works duty a, then it is *likely* that the employee works duty b on the next day. Positive rules indicate *desirable* combinations of consecutive duties.
- Negative rules: $a \rightarrow \neg b$
  If an employee works duty a, then it is *unlikely* that the employee works duty b on the next day. Negative rules indicate *undesirable* combinations of consecutive duties.

Typically, in association rule mining the user must define a *minimum support* and a *minimum confidence* threshold to create rules. Since these thresholds affect the performance of the algorithm, it is important to determine a suitable threshold. However, in practice this is difficult. To overcome this problem, Lin and Tseng (2006) propose a method that formulates relevant association rules without defining a threshold for the *minimum support*. According to Lin and Tseng (2006) a positive association rule should be created if an item set satisfies the following three definitions:

- *Automated support threshold*
  An item set is frequent if the *support* of the item set is larger than or equal to the lowest value of the support of the items included in the item set, see Equation 4.37.

$$support(a \rightarrow b) \geq min\{support(a), support(b)\} \tag{4.37}$$

- *Strong association rules*
  Moreover, the confidence of an item set must exceed the *minimum confidence*, see Equation 4.38. Although this method relieves the user from defining a minimum support threshold, the user is still required to determine a minimum confidence threshold. We apply the algorithm using different thresholds to determine the best value for the threshold.

$$confidence(a \rightarrow b) \geq Minimum\ confidence\ desirable \tag{4.38}$$

- *Interesting association rules*
  An association rule is considered as interesting if the *lift* is greater than 1, indicating a positive correlation between the consecutive duties, see Equation 4.39. A greater value of the lift indicates a stronger correlation between the consecutive duties.

$$lift(a \rightarrow b) > 1 \tag{4.39}$$

The same definitions hold for infrequent item sets, if the Equations 4.40, 4.41 and 4.42 are satisfied, a negative association rule is created. These positive and negative association rules are used to identify *desirable* and *undesirable* combinations of consecutive duties.

$$support(a \rightarrow \neg b) \geq min\{support(a), support(\neg b)\} \tag{4.40}$$

$$confidence(a \rightarrow \neg b) \leq Maximum\ confidence\ undesirable \tag{4.41}$$

$$lift(a \rightarrow b) < 1 \tag{4.42}$$

Summarizing, we use historical data to measure the frequency that a shift with duty a was followed by a shift with duty b. Subsequently, association rule mining is used to create both positive and negative association rules. These rules indicate if it is likely or unlikely that a shift with duty a is followed by a shift with duty b on the next day. Finally, these rules are used to define desirable and undesirable combinations of consecutive duties.

4.3.2 Free weekends

Previous research performed in collaboration with ORTEC explores data mining techniques to identify the preferred number of free weekends. This research proves that using association rule mining, the preferences for the number of free weekends can be identified in artificially created shift schedules (Hassan, 2019). In this approach, all possible combinations of free weekends n and periods p are enumerated. Consequently, association measures are calculated for every combination of n and p. Based on these association measures, the strongest association rule is selected and represents the preference of the customer. The following steps are taken to identify the preferred number of free weekends:

1 **Create item sets**
In historical data, we measure the frequency that employees have n free weekends in a period of p weeks. We create an item set for every combination of n and p, considering the following preferences and periods:

- The preferred free weekends, n, is smaller than the period p
  To fulfill demand, we assume that it is not reasonable to wish that employees have all weekends free in a period. Therefore, the number of preferred free weekends, n, must be smaller than the period p.
- Periods, p, from 2 to 8 weeks are included
  From practice, we know that customers do not consider a period longer than 8 weeks for this constraint type. Furthermore, a period length equal to 1 week is removed from the analysis, as the only possibility is the combination of 0 free weekends in 1 week; i.e. the organization wishes that employees never have a free weekend.

For this constraint type, the order of the items in the item sets is irrelevant, resulting in 28 different item sets.

2 **Calculate association measures**
Next, we calculate association measures. First, we calculate the *support* for each period p. The *support* of p is the fraction of observations containing p weeks compared to all

observations N, see Equation 4.43. As described before, the number of free weekends n should be smaller than the period p, therefore, n is measured up to $p-1$. Second, we calculate the *confidence* for each combination of n and p, which is the conditional probability that the period is equal to p weeks if given that an employee has n weekends off, see Equation 4.44. Finally, Hassan (2019) suggests measuring the *conviction* for all combinations of n and p, which is a measure to indicate the correlation between items, see Equation 4.45.

$$support(p) = \frac{\sum_{n=0}^{p-1} x_{n,p}^{weekends}}{\mathrm{N}} \tag{4.43}$$

where N is the total number of observations

$$confidence(n \rightarrow p) = \frac{x_{np}^{weekends}}{\sum_{n'=0}^{p-1} x_{n'p}^{weekends}} \tag{4.44}$$

$$conviction(n \rightarrow p) = \frac{1 - support(p)}{1 - confidence(n \rightarrow p)} \tag{4.45}$$

3 **Define preference**
The conviction is a measure to indicate a correlation between two items. The combination of n and p with the highest conviction have the highest correlation in historical data and represent the preference of the customer (Hassan, 2019).

Summarizing, we measure the frequency that employees have free weekends in a given period in the historical data. Using association rule mining, we detect which combination of *number of free weekends* and *period length* has the strongest correlation. This combinations represents the scheduling preference of the customer and represents the soft constraint *free weekends*.

4.3.3 Series length

The constraint *series length* is defined as follows: *A series of shifts should have a minimum length of [MinLength] and a maximum length of [MaxLength]. This constraint holds only for employees with a contract of minimal [Minimum hours] and maximal [Maximum hours] hours.* As described before, identifying the parameters of this soft constraint involves two objectives. First, we aim to identify for which group of employees the soft constraint holds based on their contract hours. Different preferences may exist for employees working part-time and employees working full-time. For planners, it is difficult to determine these employee groups. Question such as 'Which contract hours are considered as full-time?', 'Which contract hours are considered as part-time?' and 'Do we have different preferences for part-time and full-time employees?' arise. Therefore, we use historical data to cluster the contract hours based on the observed distribution of series length, these clusters are called *employee groups*. Second, we aim to identify the preferred series length for each defined employee group.

As described in Chapter 3, cluster analysis is a tool to divide data objects into different clusters. Cluster analysis can be used to achieve both objectives: i) Dividing employees over employee groups, and ii) Define the boundaries of the preferred series length. In the remainder

of this section, we explain how a nested cluster analysis can identify the soft constraint *series length*. Figure 4.9 shows a schematic overview of this process.

1 **Measure the distribution of shift series length**
We split the available data set in several data sets based on the contract hours of the employees. Subsequently, we create a histogram of the observed series length in the historical schedule per type of contract hours. The x-axis represents the series length l and the y-axis the observed frequency of this length in the historical schedule. Figure 4.9 *step 1* shows an example of the result of conducting this step. In this example, 6 unique types of contract hours exist in the department.

2 ***K*-Means algorithm contract hours**
We aim to create one or multiple employee groups based on the distributions found in the first step. The objective is to assign contract hours with a similar distribution of series length to the same employee group. K-Means clustering creates different clusters based on the observed distributions.

To apply K-Means clustering, the data must be standardized. First, the mean and the standard deviation of the series length for each type of contract hours are calculated. Subsequently, each element is standardized by subtracting the mean and dividing the result by the standard deviation.

As described in Chapter 3, we must overcome three difficulties before K-Means clustering can be applied.

- *The number of clusters, K, must be determined*
  In this case, the number of clusters K represents the number of employee groups we create. It is hard to determine the number of employee groups needed. The literature suggests using the elbow method to overcome this problem. When the number of clusters increases, the total within-cluster variation decreases, i.e. the observed distributions of series length assigned to one employee group show fewer dissimilarities. However, this effect decreases when the number of clusters increases. At some point, increasing the number of clusters only has a small effect on the within-cluster variation. The elbow method is a heuristic to detect the turning point in the improvements. According to the literature, this turning point indicates the right number of clusters K for the algorithm.
- *The results depend on the randomly initially chosen centers*
  As suggested in the literature, we apply the K-Means algorithm multiple times using different randomly chosen centers. The best solution found is selected as final solution.
- *Sensitive to outliers*
  As described in Section 4.2.3, the upper bound of the soft constraint is set to 8. Therefore, all observations longer than 8 are considered as outliers.

The output of this step is several clusters, each cluster includes one or multiple contract hours, see Figure 4.9 *step 2*. In this example, the data is divided into three clusters. Each cluster represents one employee group.

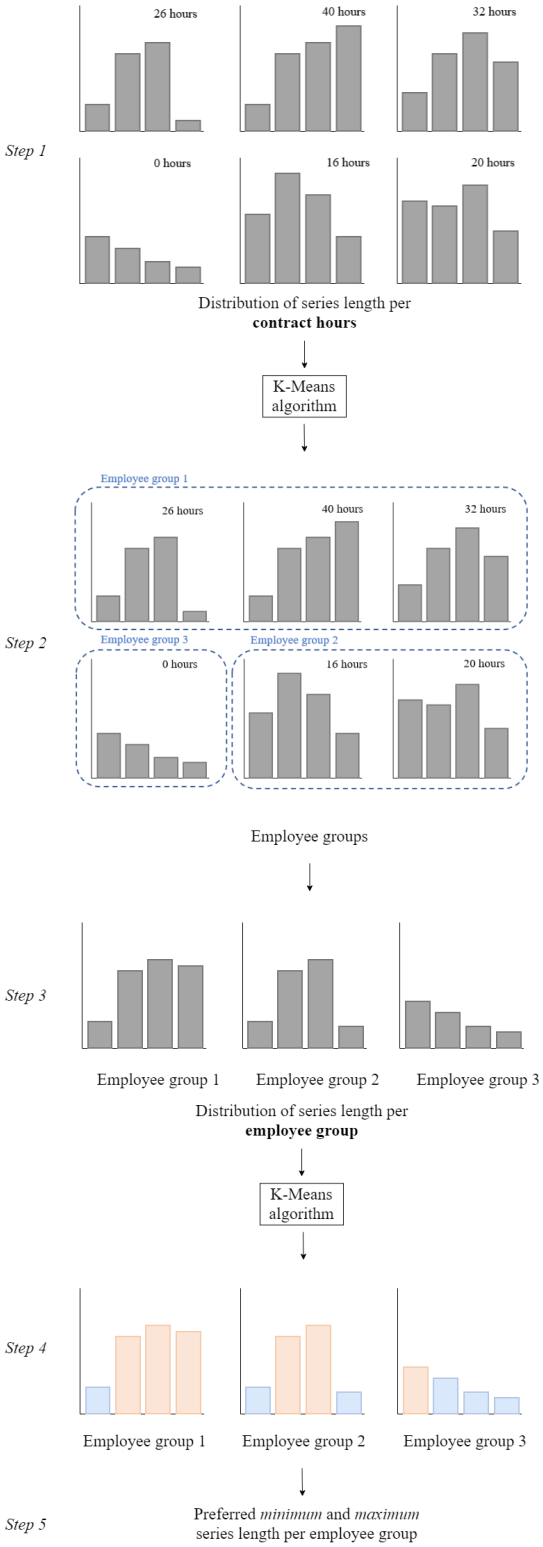


Figure 4.9: Schematic overview the algorithm to identify the preferred series length

3 **Measure series length per employee group**

Once the clustering algorithm created employee groups, the distribution of the observed series length for each employee group, $x_{g,l}^{SeriesLength}$, can be calculated. Figure 4.9 *step 3* shows the output of this step.

4 ***K*-Means algorithm employee groups**

At this point, the first objective, creating employee groups, is reached. Subsequently, we aim to determine the preferred series length for each of the employee groups. We again use a K-Means algorithm, however, now the objective is to cluster the series lengths into different clusters based on their frequency. To apply the K-Means algorithm, we apply the procedure as in step 2.

After applying this step, the series lengths l are divided into one of multiple clusters, see Figure 4.9 *step 4*. Each color represents a unique cluster of series lengths. In this example, the series lengths of each employee group are divided into two clusters.

5 **Define boundaries preferred series length**

We use these clusters to define the boundaries of the preferred series length. For each of the clusters, a cluster mean can be calculated, which is the average of the observed frequencies of the items included in the cluster. Since we assume that the preferred series length occur relative frequently, we use the cluster with the highest cluster mean to define the preferred series length: the lowest and the highest length l within this cluster represent the preference for the *minimum* and *maximum* series length, respectively. In Figure 4.9 *step 4*, for employee group 1 we observe the following two clusters: the first contains length 1 and the second cluster contains lengths 2,3,4. We observe that the second cluster has the highest cluster mean. Hence, in this example, the preferred series length for employee group 1 has a minimum of 2 and a maximum of 4.

Summarizing, the objective of identifying the soft constraint *series length* is two-fold. First, we identify employee groups for which the preference holds based on the observed distribution of series length in historical data. Second, we identify the preferred series length for each of these defined employee groups. We achieve this by applying a nested K-Means algorithm. The output is one or multiple employee groups and the preferred series length for each of these groups.

4.3.4 Distribution of night shifts

The objective of identifying the preferred distribution of night shifts is two-fold. First, we identify the series length for night shift series. Consequently, we use this information to determine the preferred number of night shifts an employee should work per period. The remainder of this section describes the approaches to identify the parameters of both soft constraints:

Preferred series length of night shifts

To identify the preferred series length of night shifts, the same data technique can be applied as for the constraint type *series length*. Customers defined the preferred series length of night shifts for all employees regardless of their contract hours, therefore, we use a simplified version of the

procedure for the constraint *series length*. The following steps describe the process for identifying the preferred series length. For a detailed description of the process we refer to Section 4.3.3.

1 **Determine** $x_l^{SeriesLengthNight}$
As described in Section 4.2.4, we assume that organizations do not find it desirable to let employees work more than 5 night shifts in a row. For this reason, we analyze the data up to a series length l of 5.

2 **Cluster the series lengths**
This step involves the same procedure as described in section 4.3.3 step 4.

3 **Define preferences series length**
This step involves the same procedure as described in section 4.3.3 step 5.

Summarizing, a K-means clustering identifies the preferred series length of night shifts for all employees within a department.

Number of night shifts

We aim to identify the frequency that an employee should work a series of night shifts. Subsequently, this information can be used to define the number of night shifts an employee should work in a period. As described in Section 4.2.4, to deal with the distribution of the night shifts better in the future we recommend ORTEC to define this soft constraint on a group level [1].

However, we cannot use the historical schedules of the customer to identify this soft constraint for the following reasons:

- *Limited amount of data*
  The available amount of historical data is limited for the number of night shifts worked. At the customer cases the demand for night shifts is small compared to the number of employees qualified to work these night shifts. Resulting, employees work a series of night shifts very occasional. As a result, the number of observations is limited. Even if we would have access to a longer period of historical data, it is questionable if the historical schedule of several years ago represents nowadays preferences.

- *High variation in the data*
  The available data shows much variation. Figure 4.7 shows the pattern for one employee of working night shifts. This figure shows that the number of weeks between a series of night shifts varies between 0 and 15 weeks.

Summarizing, due to a limited amount of data and a high variation within the data, we are not able to identify the preferred number of night shifts the employees should work in a given period based on historical shift schedules. Therefore, we aim to find an alternative approach to identify this soft constraint. We propose the following method to identify the constraint *number of night shifts*:

[1] With group level we mean a group of employees within a department that has a similar preference for working night shifts.

1 **Create three employee groups**

First, we gather the number of night shifts worked per employee in the past six months. Since the personal circumstances of employees may change over time, we include a relatively short period of the historical schedule. Based on this data, we create three groups of employees: i) Employees who prefer to work less than the average number of night shifts, ii) Employees who prefer to work the average number of night shifts and iii) Employees who prefer to work more than the average number of night shifts. We use a 30% deviation from the average number of night shifts to define the groups of employees. Figure 4.10 shows an example of these employee groups. Table 4.4 *step 1* shows the output of the first step: the number of employees included in each group.

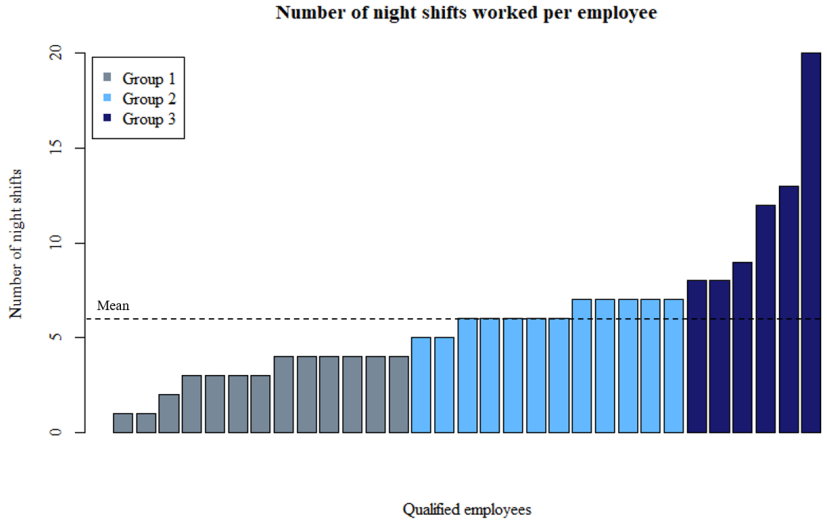


Figure 4.10: Customer A1: Number of night shifts worked per employee divided into three employee groups based on the number of night shifts worked.

2 **Measure the percentage of night shifts worked**

Next, we measure in the historical data which percentage of night shifts was covered by each of the employee groups in a period of 6 months. We assume that in the future each employee group is responsible for this percentage of night shifts. Table 4.4 *step 2* shows that each employee group is responsible to cover a different percentage of the night shifts.

3 **Calculate demand number of night shifts per employee group**

In this example, the weekly demand for night shifts is 7, see Table 4.3. Based on the percentage measured in step two and the weekly demand for night shifts, we calculate the number of night shifts each employee group should work per week. Subsequently, we calculate the number of night shifts each employee in this group is supposed to work, see Table 4.4 *step 3*.

4 **Calculate the frequency of night shift series**

Since we know that employees preferably work night shifts in a series, we calculate the frequency that an employee should work a series of night shifts. In this example, the preferred series length lies for example between 2 and 3, see Table 4.3. Table 4.4 *step 4*

shows the output of the approach. This approach results in the following formulation of the constraint: *Employees belonging to group [Group number] should work a series of night shifts every [Minimum number of weeks] to [Maximum number of weeks].*

Table 4.3: Information Night Shift Demand

Demand night shifts per week	7
Minimum series length night	2
Maximum series length night	3

Table 4.4: Customer A1: Approach to identify number of night shifts based on business knowledge

		Group 1	**Group 2**	**Group 3**	**Department**
step 1	Number of employees	13	12	6	31
step 2	Number of night shifts worked	40	69	75	185
	Fraction of night shifts worked	0.22	0.41	0.38	1.00
step 3	Night shifts per week per group	1.51	2.84	2.65	7.00
	Night shifts per week per employee	0.12	0.24	0.44	0.22
step 4	Minimum number of weeks	17	8	5	9
	Maximum number of weeks	26	13	7	13

Since we recommend ORTEC to introduce a different formulation of the constraint *number of night shifts*. We cannot compare the output of the approach we propose to the current constraints of the customers. Therefore, this type of constraint is not treated in *Chapter 5 Results*. In *Chapter 6 Conclusion and discussion* we reflect on this approach and provide ideas for further research on the distribution of night shifts.

4.4 Performance evaluation

The objective of this research is to identify scheduling preferences in historical data. These scheduling preferences can be used in the implementation phase of the Optimizer. Databases of three customers of ORTEC who currently use the Optimizer have been collected. At these customers, the Optimizer is currently implemented. In other words, we know their scheduling preferences. The soft constraints and their parameters defined by these customers can be found in Appendices A, B, D and C. Figure 4.11 shows the timeline of the historical data available of these customers. To determine to what extent data mining techniques can identify the parameters of each of the categories of soft constraints, we apply the methods proposed in Section 4.3. As indicated in the figure, only the historical data up to the moment that the customer started using the Optimizer is considered as relevant. After this point, the patterns in the shift schedules may be affected by the Optimizer.

We apply the method on the relevant historical data of the customers to identify the scheduling preferences. When applying this method, we pretend that we do not know the real scheduling preferences of these customers. Since we assume that the preferences as indicated by ORTEC's

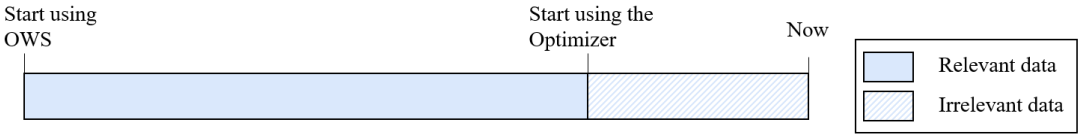


Figure 4.11: Timeline historical shift schedule. Relevant data includes the period that the customer started using OWS, up to the moment that the Optimizer is implemented.

customers are the actual preferences, these scheduling preferences are used as test data. After applying the method we propose, we compare the preferences as indicated by the method to the test data, i.e. the preferences as defined by the customers. In this way, we evaluate to what extent the method can identify the scheduling preferences using both historical data and domain knowledge.

The remainder of this section describes evaluations metrics to evaluate the performance of the method we propose for each of the categories of soft constraints.

4.4.1 Consecutive duties

A matrix with test data is created for each customer: $Test_{a,b}^{Combi}$. The constraint type consecutive duties combines three soft constraints from OWS: i) *Avoid combination*, ii) *Change in begin time*, iii) *Same duty*. Each of these currently used soft constraints must be translated to our formulation of the constraint: *consecutive duties*. The first constraint, *avoid combination*, is translated by assigning a positive value to all combinations of duties mentioned in this constraint. To translate the second constraint, *change in begin time*, all combinations of duties that do not satisfy this constraint get a positive value. Finally, to translate the constraint *same duty*, all combinations with the same duty types receive a negative value, which indicates a desirable combination of duties.

Consequently, we create a confusion matrix comparing using the identified value, $C_{a,b}^{combi}$, and the actual value, $Test_{a,b}^{Combi}$, of the soft constraint. The structure of the confusion matrix is shown in Figure 4.12. This confusion matrix allows us to calculate several evaluation metrics as explained in Section 3.3.1.

		Identified preference			
	Classes	*Desirable (-1)*	*Indifferent (0)*	*Undesirable (1)*	Total
Actual preference	*Desirable (-1)*				Total desirable
	Indifferent (0)				Total indifferent
	Undesirable (1)				Total undesirable
	Total	Total identified desirable	Total identified indifferent	Total identified undesirable	Total number of combinations

Figure 4.12: Example confusion matrix with three classes: desirable, indifferent and undesirable

4.4.2 Free weekends

As described in Section 4.1.2, the constraint free weekends consists of two variables: i) The number of free weekends n and ii) The period p. We aim to compare the identified preference based on historical data with the preference indicated by the customer. To compare these preferences, we standardize the preference by taking the ratio of $\frac{n}{p}$, which represents the percentage of free weekends an employee preferably has. In this way, we can compare the identified preference with the actual preference of the customer.

As described in Section 4.1.2, the deviation of the number of free weekends in squared. By squaring the deviation, larger deviations are considered to be more severe. However, in this case, we compare two fractions, resulting in a deviation between 0 and 1. Since this case the error lies always between 0 and 1, squaring the deviation would not give the desired effect. To overcome this problem, we use the *mean absolute error* to compare the identified and actual preference, see Equation 4.46.

The constraints *free weekends* defines the *minimum* number of free weekends an employee preferable has. However, an overestimation of the number of free weekends is still considered as an error for the following reasons:

- An overestimation of the number of free weekends could result in more weekends where employees must work both weekend days than preferred.
- The Optimizer prioritizes assigning extra vacant shifts over respecting a soft constraint. Therefore, the way weekends shifts are distributed affects to what extent the Optimizer can respect the soft constraint *preferred series length*. An overestimation of the number of free weekends results in more full weekends with work [2]. Especially for full-time employees, working full weekends affects the number of shifts these employees work in a row.

$$MAE = |\frac{n_{actual}}{p_{actual}} - \frac{n_{identified}}{p_{identified}}| \tag{4.46}$$

4.4.3 Series length

We aim to identify the preferred minimum and maximum series length using historical data and domain knowledge. After applying the method on the customer cases, we compare the identified preferences to the preferences as defined by the customers. As described in Chapter 2, this constraint has quadratic costs in OWS. By squaring the difference, larger deviations from the preference are considered to be more severe. For this reason, we use the MSE to evaluate the performance of our method.

Figure 4.13 shows some examples where the preference of the customer, the actual preference, is compared to the identified preference. The first example is the desired situation, in which the identified preference is equal to the actual preference. In this example, the error is equal to zero. In the second example, the lower bound of the identified preference deviates 1^2 from the actual preference. In the third example, the lower bound of the identified preference deviates 2 from the actual preference, squaring this deviation results in a squared of 4. In the fourth situation, both

[2] With full weekends we mean weekends where employees work both saturday and sunday.

the lower and upper bound of the identified preference deviate with 1 from the actual preference, this results in a squared of 2 ($1^2 + 1^2$).

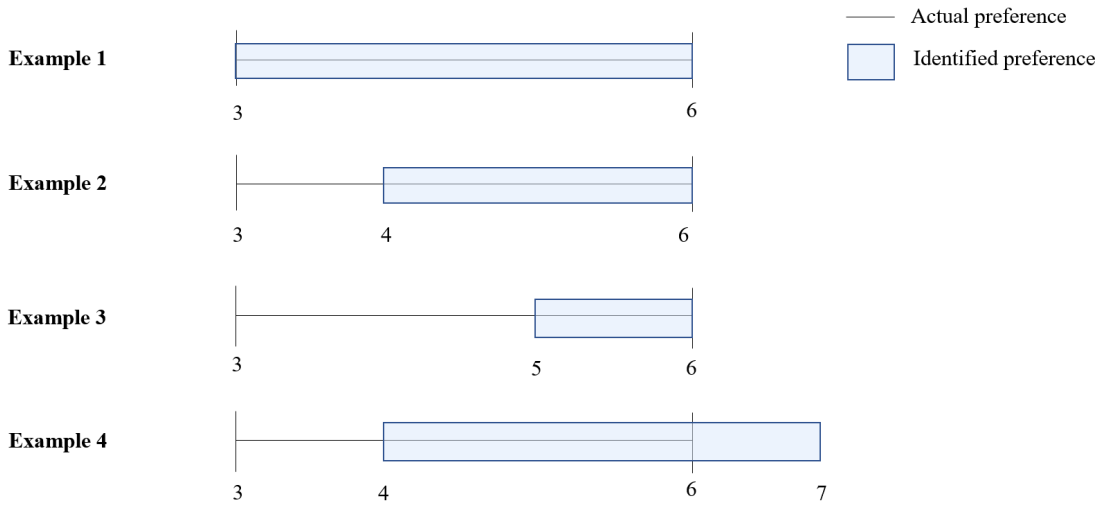


Figure 4.13: Example comparing identified series length to actual series length

As described in Section 4.3.3, we identify the preferred series length for each employee group. A wrong indication of the preference for a large employee group is considered to be more severe compared to a deviation for an employee group including few employees. Therefore, the MSE will be adjusted to the number of employees included in an employee group. Since we evaluate the identified preference for each type of contract, we evaluate both objectives of the algorithm: i) Assigning employees to the right employee group and ii) Identifying the preferred series length for each of these groups.

Equation 4.47 shows the formula for the squared error for each type of contract hour. The MSE takes the average squared error for all employees. Equation 4.50 shows the calculation for the MSE.

$$SE_{ContractHours} = NEmployees_{ContractHours} \quad * \quad (Deviation_{Min} + Deviation_{Max}) \tag{4.47}$$

Where:

$$Deviation_{min} = (Identified_{min} - Actual_{min})^2 \tag{4.48}$$

$$Deviation_{max} = (Identified_{max} - Actual_{max})^2 \tag{4.49}$$

$$MSE = \frac{1}{\sum NEmployees_{ContractHours}} \quad * \quad \sum SquaredError_{ContractHours} \tag{4.50}$$

4.4.4 Distribution of night shifts

The category distribution of night shifts contains two constraint types: i) The preferred series length for night shifts and 2) The number of night shifts employees should work in a period. The remainder of this section describes how to evaluate the performance of the method we propose.

Preferred series length night shifts

To evaluate the identified preferred series length of night shifts, we use a similar approach as for the constraint *series length*. For a detailed description of the evaluation metrics, we refer to Section 4.4.3. Equation 4.51 shows the calculation for the MSE for the constraint *preferred series length night shifts*.

$$MSE = (Deviation_{min} + Deviation_{max}) \tag{4.51}$$

Where:

$$Deviation_{min} = (Identified_{min} - Actual_{min})^2 \tag{4.52}$$

$$Deviation_{max} = (Identified_{max} - Actual_{max})^2 \tag{4.53}$$

Number of night shifts

As described in Section 4.2.4, we recommend ORTEC to reformulate their constraint to group level rather than department level. Since the current constraints used at the customers have a different structure than the constraints we propose, we cannot compare the current constraints with the identified constraints.

4.5 Conclusion

The objective of this chapter is to answer the following research question: *How can we identify the parameters of relevant soft constraints?* To answer this research question, we defined the following four sub research questions:

a *What set of soft constraints represents the scheduling preferences of ORTEC's customers?*
The customer cases included in this research, use seven types of soft constraints. We assign these soft constraints to four categories. The first category *consecutive duties* indicates the desirability of combining two duties on succeeding days. The second category *free weekends* indicates the minimum number of free weekends an employee preferably has. The third category *series length* defines the boundaries of the preferred series length per type of contract hours. The fourth category *distribution of night shifts* defines the preferred series length of night shifts and the frequency that such a series of night shifts should take place.

b *What domain knowledge can be used to identify the scheduling preferences of the customers?*
Using domain knowledge, we determine the structure and the boundaries of the soft constraints. In this way, outliers in the data can be removed and relevant patterns found in historical data can be distinguished from irrelevant patterns. We expect that this results in a better performance when identifying preferences. The relevant domain knowledge includes among others: labor laws, duty demand and employee skills.

c *Which method can we apply to identify the parameters of each of the types of soft constraints?*
For each type of soft constraint, we propose a method to identify its parameters. First,

the constraint *consecutive duties* can be identified using association rule mining to detect frequent and infrequent patterns in historical data. These patterns can be used to define desirable and undesirable combinations of duties. Second, for the constraint *free weekends*, association rule mining can be applied to define association rules. The strongest association rule represents the preference for the number of free weekends of the customer. Third, the constraint *series length* can be identified by using a nested clustering technique, which clusters the employees within a department and identifies the preferred series length for each of these clusters. Finally, the constraint *distribution of night shifts* is divided into two constraint types. The first constraint type, *preferred series length of night shifts*, can be identified using cluster analysis. To identify the second constraint type, *number of night shifts*, we cannot use the historical schedule due to a limited amount of data and high variation within the data. Alternatively, we propose a method to estimate the number of night shifts an employee should work using known variables such as demand for night shifts, number of available employees and number of night shifts worked in the past.

d *How can we evaluate the identified parameters for each of the types of soft constraints?*
The identified preferences are compared with test data that includes known preferences of the customers. For the constraint *consecutive duties*, nominal parameters must be identified. A combination of duties can either be desirable, indifferent or undesirable. The approach can be evaluated by creating a confusion matrix. For the other types of constraints, ratio parameters must be identified. Consequently, association measures for ratio parameters, such as MAE and MSE, can be used to evaluate the performance of the approaches we propose.

Chapter 5

Results

In the previous chapter, we describe four categories of soft constraints that represent the scheduling preferences of ORTEC's customers and propose a method to identify the parameters of these soft constraints. To verify the method we propose, we apply the method to eight real customer cases. In this chapter, we evaluate the results for each of the soft constraints and answer the following research question:

For each of the formulated soft constraints, to what extent can the parameters be identified based on historical data?

Each section of this chapter describes the results of one of the categories of soft constraint: Section 5.1 *Consecutive Duties*, Section 5.2 *Free weekends*, Section 5.3 *Series length* and Section 5.4 *Distribution of night shifts*. This chapter ends with a conclusion in Section 5.5.

5.1 Consecutive duties

Soft constraints that indicate a preference for consecutive duties were found at five departments from two different customers, customer A and C. Table 5.1 shows a summary of the preferences for the consecutive duties defined by these customers. Customer C4 defined only undesirable combinations of duties, all other customers defined both desirable and undesirable combinations. All soft constraints defined by these customers can be found in Appendix A.

In Section 5.1.1 we evaluate the domain knowledge used in the process. Next, in Section 5.1.2 we evaluate the performance of the method we propose to identify the constraint *consecutive duties*.

Table 5.1: Summary of customer's preferences for consecutive duties

Customer	Number of duties	Number of combinations	Number of desirable combinations (%)	Number of undesirable combinations (%)
A1	40	1600	38 (2.38%)	7 (0.44%)
C1	47	2209	4 (0.18%)	187 (8.47%)
C2	11	121	1 (0.83%)	33 (27.27%)
C3	43	1849	35 (1.38%)	135 (7.30%)
C4	36	1296	0 (0.00%)	306 (23.61%)

5.1.1 Evaluation of domain knowledge

For this constraint type, we use domain knowledge to create a forbidden duties matrix. Since the forbidden duties matrix removes irrelevant combinations of duties, it is expected that this matrix affects the performance of the algorithm positively.

Table 5.2 shows the number of possible soft constraints before and after applying known hard constraints. The effectiveness of the forbidden combinations of duties matrix varies per customer, for the Dutch customer A1, 90.5% combinations of consecutive duties are forbidden by hard constraints, while for one of the Finnish cases the number of constraints is reduced with only 8.3%. Since the customers operate in different countries, different labor laws apply. This clarifies the differences in the effectiveness of the forbidden duties matrix.

Table 5.2: Number of possible soft constraints before and after applying known hard constraints

Customer	Total number of combinations	Number of non-forbidden combinations	Reduction of number of possible constraints
A1	1600	152	90.5%
C1	2209	1524	31.0%
C2	121	111	8.3%
C3	1849	841	54.5%
C4	1296	1139	12.1%

To evaluate the effect of the forbidden duties matrix on the performance of the algorithm, we execute the algorithm with and without using the forbidden duties matrix. Table 5.3 shows the performance of the algorithm with and without using domain knowledge. This table verifies our hypothesis that applying domain knowledge has a positive effect on the performance of the algorithm.

For most customer cases, the *precision* improves slightly (between 18% and 40%) in detecting *desirable* combinations of duties. A low precision indicates that multiple combinations of duties which are identified as desirable, are actually indifferent or undesirable. Without applying the forbidden duties matrix, some forbidden combinations of duties were identified as desirable by the algorithm. This indicates that these forbidden combinations of duties were combined relative frequently in the past. This could mean that labor laws have been violated in the past, the duty demand has changed or the skill set of employees have changed.

Table 5.3: Performance of the algorithm with and without applying known hard constraints. The bold percentages indicate that applying hard constraints improves the performance of the algorithm.

		Desirable combinations			**Undesirable combinations**		
		Recall	*Specificity*	*Precision*	*Recall*	*Specificity*	*Precision*
A1	*Without*	0.71	1.00	0.71	0.43	0.17	0.00
	With	0.71	1.00	0.84	0.43	0.98	0.07
	Improvement	0%	0%	**18%**	0%	**476%**	**∞**
C1	*Without*	0.50	1.00	0.40	0.45	0.184	0.05
	With	0.50	1.00	0.50	0.45	0.91	0.31
	Improvement	0%	0%	**25%**	0%	**395%**	**533%**
C2	*Without*	1.00	0.99	0.50	0.55	0.47	0.28
	With	1.00	0.99	0.50	0.55	0.59	0.33
	Improvement	0%	0%	0%	0%	**26%**	**18%**
C3	*Without*	0.86	0.99	0.58	0.62	0.18	0.06
	With	0.86	1.00	0.81	0.62	0.86	0.26
	Improvement	0%	**1.00%**	**40%**	0%	**378%**	**333%**
C4	*Without*	-	-	-	0.63	0.35	0.23
	With	-	-	-	0.63	0.50	0.28
	Improvement	-	-	-	0%	**43%**	**22%**

As expected, the biggest improvement is found in detecting *undesirable* combinations of duties. In one of the customer cases, *A1*, the *specificity* in detecting undesirable combinations of duties improves with 476% if the forbidden duties matrix is used. The specificity is the percentage of *desirable* and *indifferent* combinations of duties that is *not* classified as undesirable. Without removing forbidden combinations of duties, these forbidden combinations are likely to be classified as undesirable. The forbidden duties matrix prevents the algorithm from classifying forbidden combinations as undesirable, this clarifies the major improvement of the *specificity*. Moreover, the *precision* improves after adding domain knowledge. The *precision* is the percentage of combinations identified as undesirable which is actually undesirable. Since the forbidden duties matrix filters for forbidden duties, less irrelevant duties are identified as undesirable, this clarifies the increase of the precision. Remarkable is the *recall*, the only measure which does not improve after applying the forbidden duties matrix. The recall is a measure of completeness and indicates to what extent the set of combinations of duties identified as undesirable is complete. Since the forbidden duties matrix does not affect the way a combination of duties is evaluated, no additional combinations of duties are identified as undesirable. Consequently, the recall is not affected by the forbidden duties matrix.

Furthermore, before association rule mining can be applied, a support and confidence threshold must be defined. The method that Lin and Tseng (2006) propose, relieves the user from setting a support threshold, however, this method still requires the user to define a confidence threshold. To determine the effect of the confidence threshold on the performance, the algorithm was executed without confidence thresholds and with varying confidence thresholds.

Increasing the *minimum confidence*, results in fewer combinations of duties being identified as desirable. The *recall*, a measure of completeness, is expected to increase if the minimum confidence increases. On the other hand, the *precision*, a measure of exactness, is expected to increase if the *minimum confidence* increases. The results from the algorithm confirm this expectation. Figure 5.1 shows the effect of the *minimum confidence* on the performance of the algorithm. The graph shows that the *minimum confidence* threshold affects the *recall* and *precision*. However, the *specificity* remains stable with a varying minimum confidence. The specificity indicates to what extent the method is able to distinguish the desirable combinations of duties from the indifferent and undesirable combinations of duties.

The F_β combines the *recall* and *precision*. The recall is the fraction of the relevant constraints that are missed and the precision is the fraction of irrelevant combinations is identified as relevant. For ORTEC is it equally important to not miss any important constraints and to identify only relevant constraints. In other words, the recall and precision are considered to be equally important. Consequently, a β equal to 1 is used. Therefore, a maximum F_1 indicates the optimal minimum confidence threshold, which is equal to 0.2 for this customer case.

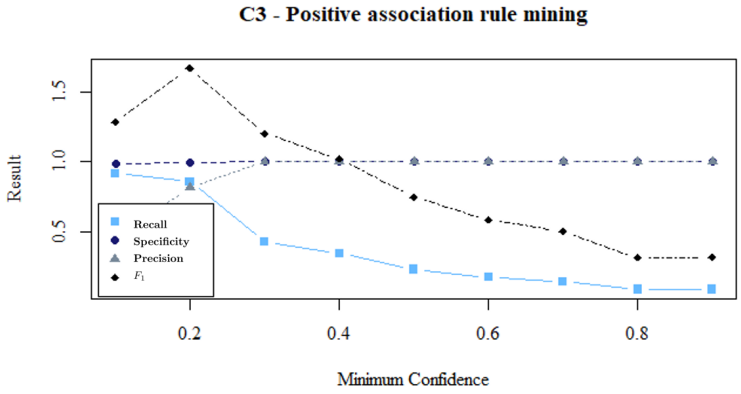


Figure 5.1: Effect of the minimum confidence thresholds on the performance of association rule mining to identify desirable combinations of duties for customer C3

Figure 5.2 shows the effect of the minimum confidence level on the F_1 in detecting desirable combinations of duties per customer case. For the customer cases A1, C1, C2 and C3 the optimal *minimum confidence* threshold equals 0.4, 0.5, 1.0 and 0.2, respectively. This result confirms the expectation that it is not possible to determine a uniform threshold that results in optimal results for all customer cases.

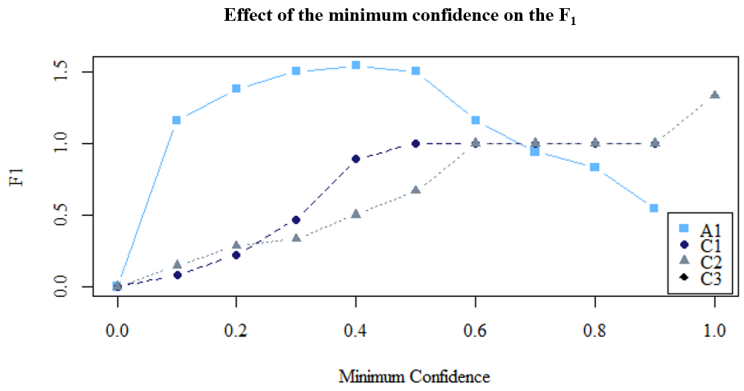


Figure 5.2: Comparison of the effect of the minimum confidence level on the F_1 measure for all customer cases for desirable combinations of duties

The same analysis was performed for undesirable combinations of duties. For the undesirable combinations, a higher *maximum confidence* threshold results in more combinations of duties classified as undesirable. Therefore, it is expected that a higher maximum confidence threshold results in a higher *recall* and a lower *precision*. Figure 5.3 shows the effect of the maximum confidence on the performance of the algorithm for customer C1. Since the recall and precision remain stable if the maximum confidence increases, these results are dissimilar to the expectation. This means that other constraints, for example, the maximum lift or the minimum support, are binding in detecting undesirable combinations of duties.

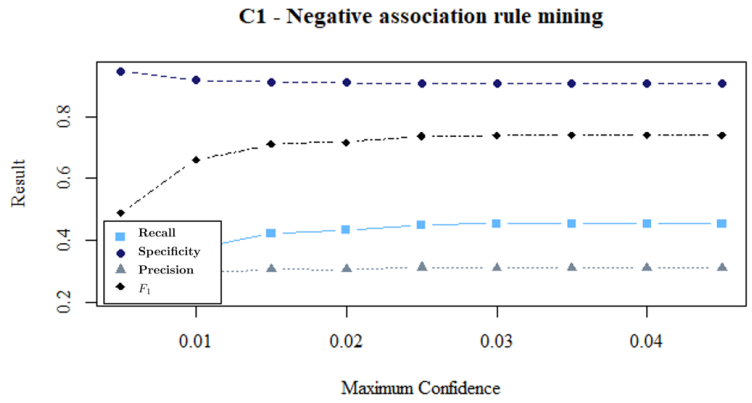


Figure 5.3: Effect of the maximum confidence thresholds on the performance of association rule mining for customer C1

Figure 5.4 shows a comparison of the effect of the maximum confidence level on the F_1 measure per customer case for the undesirable combinations of duties. The graph shows that for each customer case the F_1 reaches its peak for a maximum confidence greater than or equal to 0.1. In other words, the maximum confidence is redundant while identifying undesirable combinations of duties. Consequently, there is no need to set a maximum confidence for identifying undesirable combinations of duties.

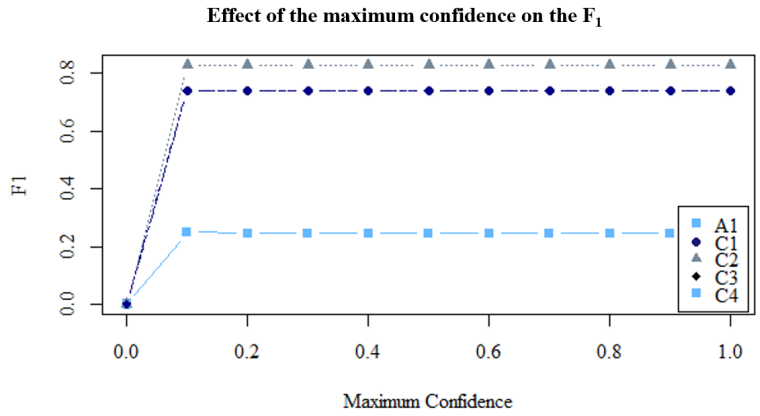


Figure 5.4: Comparison of the effect of the maximum confidence level on the F_1 measure for all customer cases for undesirable combinations of duties

Summary

- The effectiveness of applying domain knowledge varies per customer case. The differences can be clarified as the customers operate in different countries.
- The forbidden duties matrix filters out irrelevant combinations of duties, resulting in less forbidden combinations of duties are identified as *undesirable*. As a result, adding the forbidden duties matrix results in better performance, especially in detecting *undesirable* combinations of duties.
- The forbidden duties matrix does not affect the completeness of the result; the *recall* is not affected by the forbidden duties matrix.
- The performance of the algorithm for identifying *desirable* combinations of duties depends on the *minimum confidence* threshold. Since the distribution of the data differs per customer, we cannot find one optimal threshold. We recommend using a threshold of 0.5 if the distribution is unknown.
- The results of identifying *undesirable* combinations of duties are not affected by the *maximum confidence* threshold. Therefore, the maximum confidence threshold can be left out in the algorithm.

5.1.2 Evaluation results

For each customer case, the algorithm was applied using the forbidden duties matrix, its optimal minimum confidence level and without a maximum confidence level. Figure 5.5 shows the accuracy of the approach for each customer case. The *accuracy* is the percentage of combinations of duties that is classified correctly. An accuracy equal to 1 indicates that all constraints are identified correctly. The graph shows that the accuracy varies between the customer cases. For customer A1, the accuracy is close to 1, while for customer C2 and C4, the accuracy is slightly above 0.5. This means that for customer case C2 and C4 only 50% of the combinations were identified correctly. The *accuracy* evaluates the number of correctly identified combinations of duties

and does not make a distinction between undesirable and desirable combinations of consecutive duties. To compare the performance of the method for identifying desirable and undesirable combinations of duties, we use other evaluation metrics such as *recall*, *specificity* and *precision*. The remainder of this section compares the performance for mining desirable and undesirable combinations and clarifies the differences found between the customer cases.

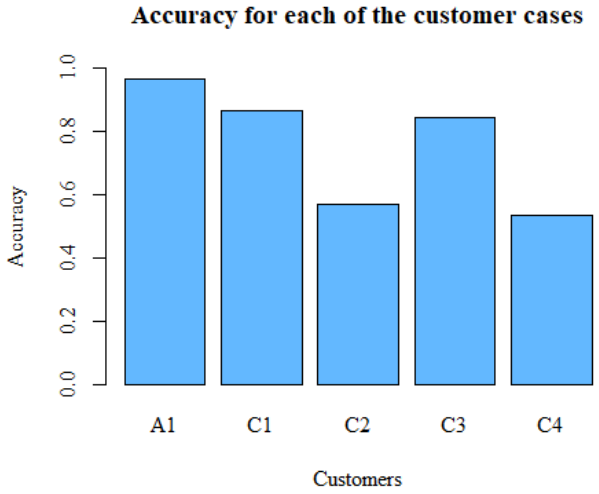


Figure 5.5: Accuracy of the algorithm for each of the customer cases

Figure 5.6 shows the evaluation metrics to evaluate the performance of detecting desirable combinations of consecutive duties per customer case. For each customer case, the *specificity* is close to 1.0. The specificity indicates to what extent the method can distinguish the *desirable* combinations of duties from the other combinations of duties. A specificity close to 1.0 means that most combinations identified either as *undesirable* or *indifferent* are indeed not perceived as *desirable* by the customer. Furthermore, the *recall* varies between 0.5 and 1.0. The recall is a measure of completeness. A recall of 1.0 means that all desirable combinations of duties have been identified by the algorithm. Whereas, a recall of 0.5 means that 50% of the most important desirable combinations of duties have been identified as desirable. Additionally, the precision varies between 0.5 and 0.85. The *precision* is a measure of exactness, in other words, the percentage of combinations identified as desirable, which is also perceived as desirable by the customer. Since the precision is not equal to 1.0, we are not sure if all combinations identified as desirable by the algorithm are indeed desirable.

The findings can be clarified by looking further into the results. Table 5.4 shows a confusion matrix for customer A1. An interesting finding from this matrix is that 4 combinations of duties were identified by the algorithm as undesirable, while the customers perceived these combinations as desirable. If we take a further look at these 4 combinations of duties, we see that in the past, these duties were never combined on two succeeding days. We assume that either the customer defined these preferences incorrectly or the customer did not succeed to comply with these preferences in the past. As described before, we assume that the preferences as defined by the customer are the actual preferences, however, the results show that the preferences as defined by the customer might be incorrect or impossible. Since preferences defined by the customer are

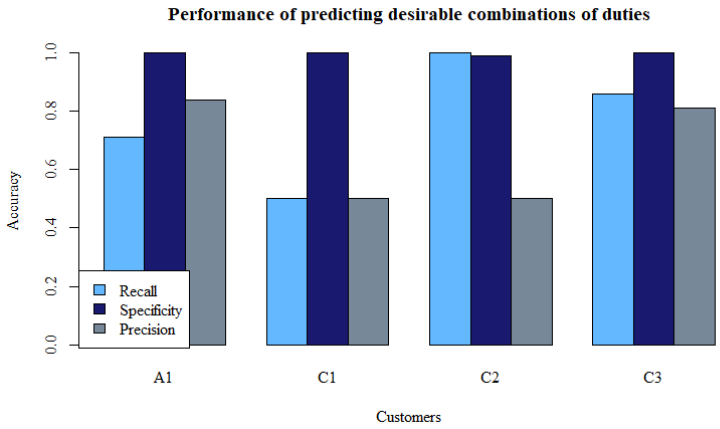


Figure 5.6: Performance of identifying desirable combinations of duties per customer case

used to evaluate the performance of the algorithm, the results may be influenced by the extent to which the customer was able to define the actual preferences and the extent to which the customer could adhere to these preferences in the past. This clarifies that the performance of the algorithm is different for each customer case.

Table 5.4: Customer A1: confusion matrix

		Identified			
		Desirable	**Indifferent**	**Undesirable**	**Total**
Actual	**Desirable**	27	7	4	38
	Indifferent	5	1515	35	1555
	Undesirable	0	4	3	7
	Total	32	1526	42	1600

Figure 5.7 shows the *recall*, *specificity* and *precision* for detecting undesirable combinations of duties per customer case. This figure shows that the *specificity* in detecting undesirable combinations is worse compared to detecting desirable combinations. This means that either desirable or indifferent combinations of duties were detected as undesirable by the algorithm. Table 5.4 shows that 35 combinations of duties perceived as indifferent by the customer, were indicated as undesirable by the algorithm. If we take a further look at these 35 combinations of duties, we see that 57.14% of these duties have never been combined on succeeding days in the past. This could indicate that other underlying hard constraints restrict that the duties in these item sets are combined on succeeding days. Subsequently, there is no need to define a soft constraint for the customer to prevent these combinations of duties. Since the Optimizer always respects hard constraints over satisfying soft constraints, this incorrect classification does not influence the output of the Optimizer. Furthermore, just as in mining desirable combinations of duties, the recall and precision vary between the customer cases. This means that we cannot say if the algorithm succeeded in identifying all undesirable combinations of duties and if all undesirable combinations of duties are indeed perceived as undesirable.

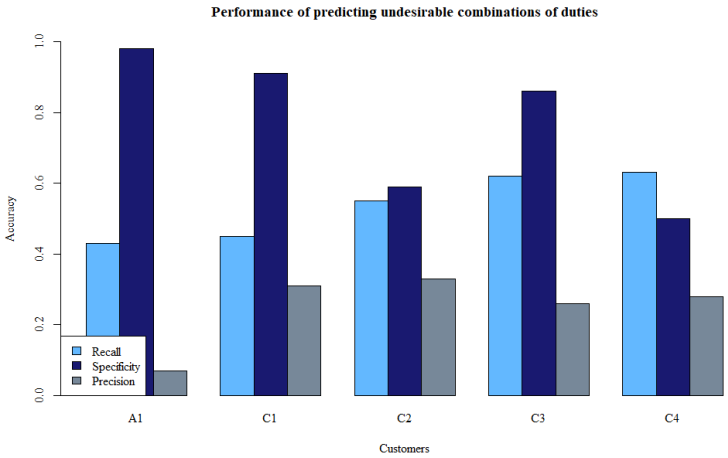


Figure 5.7: Performance of identifying undesirable combinations of duties for each customer case

Summary

- The algorithm could identify 40% to 100% of the most important soft constraints. For practice, this means that the consultant should verify with the customer whether all important preferences were found or not.
- The *precision* varies between 0.5 and 0.85 for identifying desirable combinations of duties and between 0.1 and 0.35 for identifying undesirable combinations of duties. This means that not all identified preferences are indeed a preference of the customer. Although the set of soft constraints representing undesirable combinations of duties contain many irrelevant combinations, the results show that this does not affect the quality of the schedules as these combinations never occurred in the past.
- Some of the preferences of the customers were not in line with the historical shift schedules. This means that either the customer did not succeed in expressing the scheduling preferences or the customer could not adhere to these preferences in the past. Therefore, it is hard to say if some of the soft constraints identified by the algorithm are indeed wrong or not.

5.2 Free weekends

Three different departments from customer B indicate a preference for the minimum number of free weekends. Appendix B describes the soft constraints of these customer cases. In Section 5.2.1, we evaluate the domain knowledge used in the process. Next, in Section 5.2.2, we evaluate the performance of the algorithm in identifying the constraint *Free weekends*.

5.2.1 Evaluation of domain knowledge

As described in Chapter 4, in practice, weekend shifts are not distributed equally among the employees. The extent a planner can adhere to a preference for the number of free weekends,

strongly depends on the number of weekend shifts an employee works. Since the average number of weekend shifts an employee works, varies between the employees, we select the historical shift schedules of employees facing the highest burden of weekend shifts. Table 5.5 shows the results of applying association rule mining with different selections of employees included in the analysis: i) All employees, ii) Employees working more weekend shifts than the 25th quartile, iii) Employees working more weekend shifts than the median, and iv) Employees working more weekend shifts than the 75th quartile.

For customer case B1 and B2, including all employees in the analysis results in an overestimation of the preferred number of free weekends. The constraints defines the *minimum* number of free weekends. Therefore, the organization perceives it as desirable if employees have more free weekends than indicated. This clarifies the overestimation of the number of free weekends.

In Table 5.5 the highest conviction is marked bold. The highest conviction indicates the strongest correlation. For all customer cases, the algorithm performs best when using the selection of employees which gives the highest conviction. For customer B1 and B3, this results in an MAE equal to 0, indicating a perfect estimation of the soft constraint. For customer case B2, this results in a small MAE equal to 0.03, in other words, the algorithm under- or overestimated the fraction of the preferred number of free weekends with 3%.

Table 5.5: Comparison of the MAE with different selection of employees based on the average number of weekend shifts worked. The bold numbers represent the best result(s) per customer case.

	B1				**B2**				**B3**			
	Identified	*Actual*	*MAE*	*Conviction*	*Identified*	*Actual*	*MAE*	*Conviction*	*Identified*	*Actual*	*MAE*	*Conviction*
All employees	1/2	1/3	0.17	1.27	1/2	1/3	0.17	1.21	1/4	1/4	0.00	1.34
1st quartile	1/3	1/3	0.00	1.30	3/10	1/3	0.03	1.21	1/4	1/4	0.00	1.38
Median	1/3	1/3	**0.00**	**1.31**	3/10	1/3	**0.03**	**1.22**	1/4	1/4	0.00	1.43
3rd quartile	1/4	1/3	0.08	1.27	3/10	1/3	**0.03**	**1.22**	1/4	1/4	**0.00**	**1.49**

Summary

- In two out of three customer cases, the selection of the employees affects the results of the algorithm.
- For all customer cases, the selection of employees with the highest conviction gives the lowest MAE.
- For this type of soft constraint, the MAE varies between 0.00 and 0.03, which indicates good performance.

5.2.2 Evaluation results

Table 5.5 shows that association rule mining can identify the fraction of preferred number of free weekends, given that the right employees are selected in the analysis. Table 5.6 shows the results of applying association rule mining on the data set of customer D1 in more detail. The customer defined the following preferences: employees should have at least 2 weekends off in a period of 6 weeks. According to Hassan (2019), the preference is the combinations of n and p with the highest conviction. In this case, this would results in the following preference: 1 weekend free in every 3 weeks. The fraction of free weekends is identified correctly ($1/3 = 2/6$), however,

the period is identified incorrectly. If we have a further look at the table, we see that in the second column, the preferences 2 out 6 weekends has the highest conviction. For the other two customer cases, the detailed results look similar to those of customer D1. We conclude that the algorithm was able to find the preferred fraction of weekends off, which is equal to 1/3 in this case. However, the algorithm did not succeed in identifying the right period p.

Table 5.6: Conviction for all combinations of n number of weekends free in a period of p weeks for customer B1. The customer finds 2 weekends free in a period of 6 weeks desirable.

p\n	**1**	**2**	**3**	**4**	**5**	**6**	**7**
2	1.10						
3	**1.31**	0.86					
4	1.29	0.97	0.89				
5	1.19	1.11	0.94	0.96			
6	1.02	**1.22**	1.01	0.98	0.99		
7	0.90	1.21	**1.10**	1.01	1.00	1.00	
8	0,80	1,04	0.81	**1.15**	**1.18**	**1.07**	**1.01**

Summary

- The algorithm can identify the fraction of free weekends an employee should have with a very small MAE.
- The algorithm did not succeed in identifying the correct period the constraint holds for. Therefore, human interaction with the planner is required to determine for the correct period length.

5.3 Series length

The preference for the series length is found at 7 departments from organization B and C. Appendix C describes the soft constraints of these customer cases. As described in Section 4.3.3, we propose to apply a nested K-Means algorithm to identify the preferred series length. In Section 5.3.1 we evaluate the domain knowledge used in the algorithm. In Section 5.3.2 we evaluate the performance of the method we propose.

5.3.1 Evaluation of domain knowledge

Before the algorithm can be applied, we have to overcome three difficulties: i) The number of clusters K, ii) The number of iterations and iii) Detect outliers. Some of the assumptions made to overcome these difficulties are based on domain knowledge. In this section, we evaluate the assumptions made in each of these difficulties.

The number of clusters K

In order to use the K-Means algorithm, the number of clusters K must be determined. As suggested in the literature, we apply the elbow method to select the right number of clusters K. Therefore, we expect that applying the elbow method results in the best performance of the

algorithm. However, applying the elbow method to select the right number of clusters did not have the desired effect. In most customer cases, according to the elbow method, the right number of clusters is equal to 2. This means that the series lengths l are divided into two clusters; a cluster including the lengths with a high frequency and a cluster including the lengths with a low frequency. As a consequence, in many cases, the range of the identified preferred series length was much smaller than the preferred series length as indicated by the customers.

For this reason, we deviate from what the literature suggests. Instead of using the elbow method to define the number of clusters K, we use a predetermined number of clusters. Table 5.7 compares the MSE for using the elbow method and using a predetermined number of clusters K. The MSE represents to what extent the identified preferred series length complies with the preferred series length as perceived by the customer. The table shows that in 6 out of 7 the elbow method does not give the best results. For these customer cases, the best results are obtained when using 4 or 5 clusters. Later in this section, we clarify the difference between the performance of the K-means algorithm with the elbow method and a predetermined number of clusters.

Table 5.7: MSE for identifying the preferred series length for each customer using different number of clusters. The bold numbers represent the best results for that customer case.

Customer	**MSE**						
	Elbow	$K = 2$	$K = 3$	$K = 4$	$K = 5$	$K = 6$	$K = 7$
B1	4.82	4.82	6.18	**4.55**	**4.55**	**4.55**	5.00
B2	8.00	8.00	8.00	8.00	**4.00**	5.00	5.00
B3	**1.68**	**1.68**	**1.68**	8.00	8.00	8.00	8.00
C1	1.29	8.95	1.11	**0.16**	0.93	1.10	2.34
C2	7.95	7.95	1.79	**0.90**	0.93	1.10	2.34
C3	9.93	9.93	1.93	**0.07**	**0.07**	**0.07**	1.07
C4	2.31	5.20	2.01	2.03	**0.16**	**0.16**	2.49

An interesting case is customer B3, the table shows that only for this customer case, the elbow method gives better results than using a predetermined number of clusters. Figure 5.8 shows the observed distribution of series length for this customer for employees working between 30 and 48 hours. The customer perceives a series length between 3 and 5 shifts as preferred. The figure shows that these series lengths indeed have a higher frequency than other series lengths. In general, using a low number of clusters results in a small range of series lengths identified as desirable, only lengths with the highest frequency are identified as desirable. Moreover, using a higher number of clusters results in a wider range of series lengths identified as desirable. Compared to other customer cases, for customer case B3 the range of preferred series lengths is small: only three, four, or five shifts in a row are perceived as desirable. This clarifies that using a low number of clusters, gives better results for this specific customer case.

To confirm this, we take a further look on customer case C1, for this customer case, the MSE decreases with 1.13 and thereby the performance improves with 87.6% if 4 clusters are used compared to using the elbow method. Figure 5.9 shows the observed series length for customer C1 including their preferences. This graph shows a strong peak for series length of 5. Due to

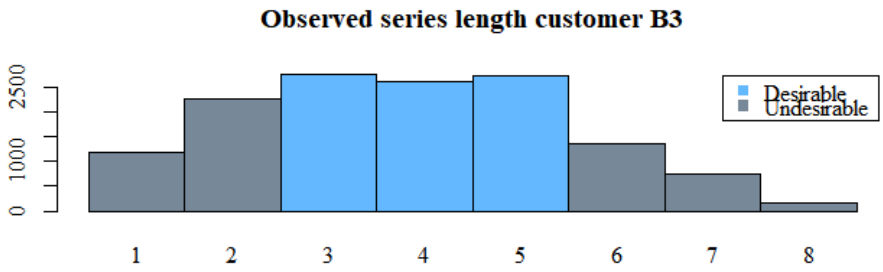


Figure 5.8: Observed distribution of series length for customer B3 for employees working between 30 and 48 hours. The preferred series length as defined by the customer lies between 3 and 5 shifts.

this strong peak, using the elbow method results in two clusters; the first cluster contains only length 5 and the second cluster contains all other lengths. Consequently, only a series length of 5 is identified as desirable. Increasing the number of clusters results in more lengths being identified as desirable. Therefore, in this case, increasing the number of clusters results in a better estimation of the preferences of the customer.

Figure 5.9: Observed distribution of series length for customer C1 for employees working between 18 and 48 hours. The preferred series length as defined by the customer lies between 2 and 5 shifts.

Number of iterations

Since the results of the K-means algorithm may depend on the randomly chosen start solution, we performed 25 iterations of the algorithm using different random starting solutions. We observe that in all customer cases, after 2 iterations the algorithm did not find any improvements.

Detect outliers

Using domain knowledge, we set the boundary of this soft constraint to 8. Consequently, all observations with a series length longer than 8 are identified as outliers. Since the results of the

K-Means algorithm depend on outliers, we evaluate if the assumption made is correct.

Another way to determine the upper bound of the soft constraint is to detect outliers in the data. Literature suggests several methods to deal with outliers (Kwak and Kim, 2017). A frequently used method is a boxplot. The Interquartile Range (IQR) is used to detect the outliers; observations outside the boundaries of $1.5 * IQR$ are identified as outliers. Figure 5.10 shows the boxplots for the observed series length at the customer cases. The boxplots show that the distribution of the series lengths differs per customer case; for customer C series lengths longer than 8 are considered as outliers, while for customer B2 series lengths longer than 3 are considered as outliers.

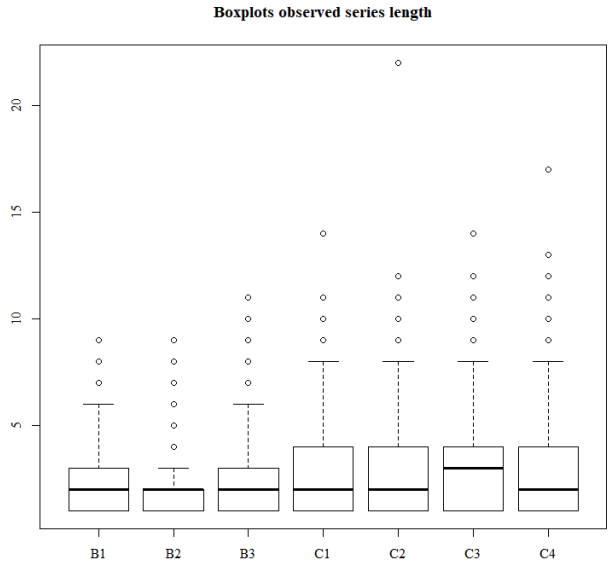


Figure 5.10: Boxplot observed series length for all employees per customer case

Table 5.8 shows that in 6 out of 7 cases, using the outliers detected in the data as the upper bound of the soft constraint gives the lowest MSE. Only for B2, the outliers in the data deviate from the preferences of the customer; the prefers a maximum series length of 7, while shift series longer than 3 are detected as outliers in the data. This indicates that the historical shift schedule of this customer is not in line with their preference.

Table 5.8: Comparison of the MSE using different boundaries for the soft constraint preferred series length varying between 6 and 10 shifts in a row. The bold number represent the best results per customer case.

Customer	**Outliers (1.5*IQR)**	**Upper bound series length**				
		6	*7*	*8*	*9*	*10*
B1	6	**3.19**	4.55	4.55	4.55	5.00
B2	4	8.00	8.00	8.00	8.00	8.00
B3	6	**2.45**	2.45	2.45	8.00	8.00
C1	9	0.16	0.16	**0.16**	**0.16**	0.16
C2	9	0.95	0.95	**0.90**	**0.90**	0.90
C3	9	0.07	0.07	**0.07**	**0.07**	0.07
C4	9	0.97	0.97	**0.16**	**0.16**	0.20

Summary

- In contrast to suggested in the literature, we use a fixed number of clusters K. For most customer cases, 4 or 5 clusters give the best results. However, when the customer searches for a small range of preferred series length, it is recommended to use fewer clusters.
- Only 2 iterations of the algorithm have to be applied.
- The results are affected by the upper bound of the soft constraints. Using outliers in the data to define the upper bound of the soft constraint gives better results than using domain knowledge.

5.3.2 Evaluation results

The K-Means algorithm indicates the preferred series length per type of contract. However, the customer may not have defined a preference for all possible contract hours. Therefore, we distinguish negative and positive cases. Positive cases are defined as follows: contract types for which the customer defined a preference for the series length. In negative cases, the customer did not define a preference for this contract type. In the remainder of this section, we evaluate the performance of the algorithm for identifying both positive and negative cases.

Positive cases

Furthermore, Table 5.7 shows that the performance of the algorithm varies between the customers. For customer C, the algorithm performs better compared to customer B. In customer case C1, the MSE is equal to 0.16 (if $K = 4$), this means that the squared deviation of the identified preferences from the actual preference is on average 0.16. Table 5.9 shows the results of the identified series length for customer C1 per employee group. This table shows that the algorithm could identify the preferred series length for large employee groups with an MSE equal to 0. For small employee groups, containing only one employee, the performance is worse. In this case, the MSE for small employee groups is at the highest 4. This difference in performance

could be clarified by the amount of data available for the employee groups.

Table 5.9: Identified preferred series length for customer C1 per type of contract

Contract hours	**Employee cluster**	**Number of Employees**	**Identified preference**		**MSE**		**Between Variation**	**Within Variation**
			Minimum	*Maximum*	*Minimum*	*Maximum*		
0.00	2	43	1	5	-	-	6.61	0.39
26.83	2	1	1	5	1	0	6.61	0.39
30.28	3	1	2	7	0	4	7.00	0.00
30.67	3	1	2	7	0	4	7.00	0.00
31.07	1	8	2	5	0	0	6.81	0.19
34.50	3	1	2	7	0	4	7.00	0.00
38.33	1	10	2	5	0	0	6.81	0.19
38.83	1	60	2	5	0	0	6.81	0.19

However, for some customer cases of customer B, the algorithm did not perform well. For customer case B2, the MSE is 4.00. This means that the squared deviation of the identified range compared to the actual range is on average 4.00. Customer B2 indicates that for employees working between 30 and 48 hours, the preferred series length lies between the 3 and 7 shifts. Our algorithm identified that the preferred series length for this employee group lies between 1 and 5 shifts. Figure 5.11 shows the observed distribution of the series length for customer D2 for this employee group. This figure shows that the preference of the customer is not in line with the historical shift schedule; a shift series with a low frequency in historical data, $l = 7$, is perceived as desirable, while a shift series with the highest frequency in historical data, $l = 2$, is perceived as undesirable. This could indicate the customer is not aware of their real preferences, or the customer was not able to satisfy their preferences in the past. Stating scheduling preferences is subjective in nature, therefore, it is hard to compare the results of the algorithm to the preferences stated by the customer. If we look at the distribution of the series length in Figure 5.11, we conclude that the algorithm was able to extract the most frequent series lengths from the historical data. This clarifies the difference between the performance of the algorithm for customer B and C. The preferences defined by customer C were more in line with their historical shift schedule compared to customer B.

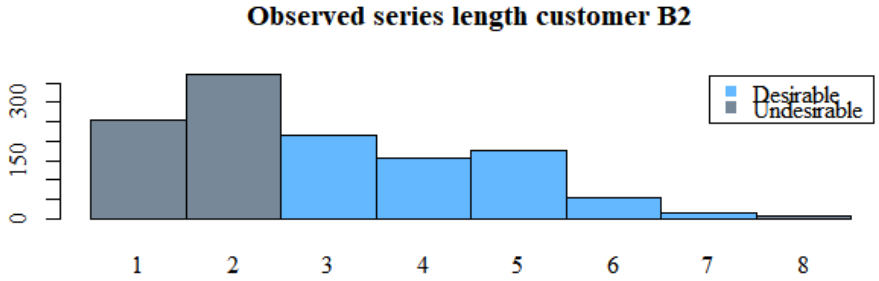


Figure 5.11: Observed distribution of series length for customer B2 for employees working between 30 and 48 hours. The preferred series length as defined by the customer lies between 3 and 7 shifts.

Negative cases

Furthermore, next to identifying the preferred series length, it is also valuable to know if the identified series length is a relevant soft constraint for this employee group or not. For example, as shown in Table 5.9, the identified preferred series length for employees with a 0-hours contract lies between 1 and 5. However, the customer did not define a preference for employees with a 0-hour contract.

If a strong preference exists for a series length, it is expected that the preferred series length has a high frequency and the non-preferred series length a low frequency. On the other hand, if a customer does not have a preference for the series length, we expect that all series lengths have an equal frequency. If we translate this to the results of the K-Means algorithm, a low between-cluster variation indicates that the elements in the clusters look similar to each other. Therefore, we expect a lower between-cluster variation for employee groups where no preference exists. Figure 5.12 shows the between-cluster variation for all seven customer cases. Every point in the graph represents a type of contract. This figure shows that there is no strong difference in the between-cluster variation of negative and positive cases. Additionally, this graph shows that the between-cluster variation varies between the customer cases, this can be clarified by the fact that every customer case has its own unique distribution of data. Since we do not know these distributions on forehand, it is not possible to determine a minimum threshold for the between-cluster variation to distinguish positive and negative cases.

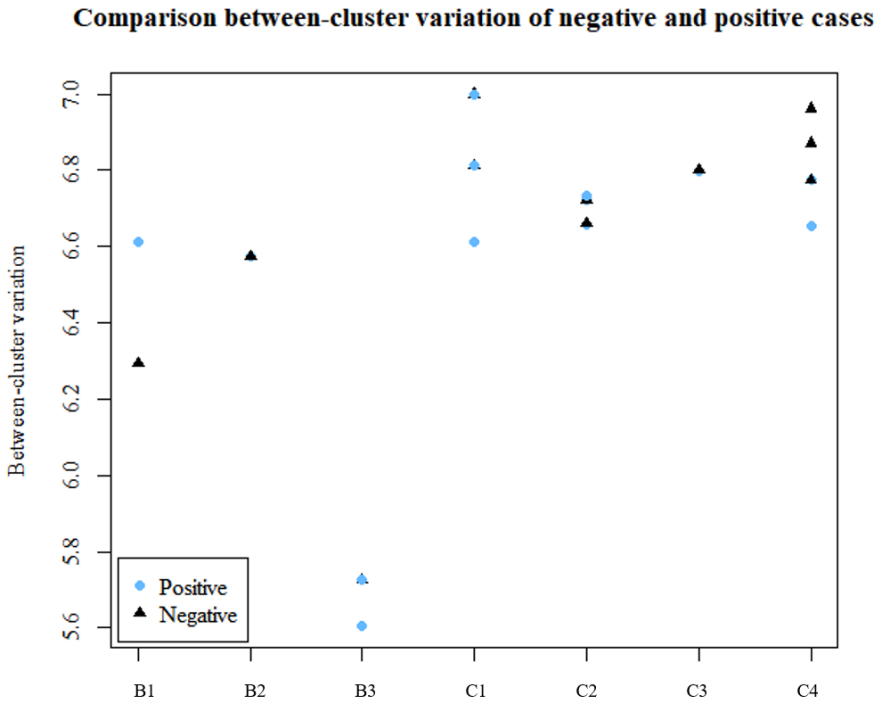


Figure 5.12: Comparison of between-cluster variation of negative and positive cases per customer case. Positive cases are employee groups where the customer defined a preference for the series length, negative cases are employee groups where the customer did not defined a preference for the series length.

Summary

- For customer C, the preference of the customer was in line with what happened in the past. Therefore, the preference could be identified with an MSE varying between 0.16 and 0.90.
- However, the preferences of customer D were not in line with the historical shift schedule, resulting in an MSE varying between 1.68 and 4.55. Therefore, in practice, the consultant should verify with the customer, if the historical shift schedule was preferred or not.
- The algorithm is not able to identify whether the preference for a series length exists for an employee group or not. Therefore, human interaction is required to evaluate if the identified preference should be translated into a soft constraint or not.

5.4 Distribution of night shifts

Two different customers, customer A and C, defined preferences for the distribution of night shifts. Appendix D describes these soft constraints for these customers. As described in Section 4.1.4 two constraint types are used to distribute the night shifts over the employees: i) The preferred series length of a night shifts series and ii) The preferred number of night shifts. As described before, we propose a different structure for the constraint *number of night shifts*. For that reason, this section only evaluates the results obtained for the constraint *preferred series length night shifts*. Section 5.4.1 evaluates the domain knowledge used to identify this constraint type. Next, Section 5.4.2 evaluates the performance of the method we propose.

5.4.1 Evaluation of domain knowledge

To identify the preferred series length of night shifts, we apply a K-Means algorithm. As described in Chapter 4, we must overcome the three difficulties to apply the K-Means algorithm. In the remainder of this section, we evaluate the domain knowledge used and decisions made for each of these difficulties.

The number of clusters K

As suggested in the literature, we use the elbow method to determine the right number of clusters K. However, as described in Section 5.3, using the elbow method did not have the desired effect while identifying the preferred series length.

Table 5.10 shows the results of using the elbow method and a fixed number of clusters K in identifying the preferred series length of night shifts. The results show that in 4 out of 5 cases, using the elbow method gives the best results. As described in Section 5.3, the elbow method results in a small range compared to the preference of the customer. However, customers define a smaller range for the preferred series length of night shifts compared to general shifts series. This clarifies the different results for identifying the preferred series length and preferred series length for night shifts specific.

Table 5.10: Results of the algorithm with a fixed number of clusters and the elbow method to determine the number of clusters. The bold numbers represent the best results for each of the customer cases.

Customer	**MSE**			
	Elbow	*k=2*	*k=3*	*k=4*
A1	**0**	**0**	1	1
C1	5	5	**1**	**1**
C2	**0**	**0**	1	4
C3	**0**	1	**0**	**0**
C4	**1**	**1**	4	4

Number of iterations

Just as for the constraint *series length*, the K-means algorithm found the best solution after 2 iterations. Therefore, 2 iterations are sufficient to obtain the best possible results.

Detect outliers

The K-Means algorithm is sensitive to outliers, and therefore it is important to carefully remove outliers. From domain knowledge, we know that customers do not prefer that employees work more than 5 night shifts in a row. In other words, we set the boundaries of the soft constraint to a maximum of 5. To verify this assumption, we apply the algorithm using different boundaries for the soft constraint. Table 5.11 shows the results of this process. We observe that in 4 out of 5 cases, the performance of the algorithm is much worse when we increase the upper bound from 5 to 8.

Only in one of the customer cases, C1, the results get worse when setting a smaller boundary. The table shows that the algorithm performs worse for this customer case compared to the other cases. Therefore, the results of this customer case are not representative for the other cases. In Section 5.4.2, we clarify the bad performance of the algorithm for customer C1.

Based on these observations, we conclude that adding domain knowledge is valuable in identifying the preferred series length of night shifts. Moreover, we verify that the domain knowledge was correct, in most customer cases a maximum series length of 5 gives the best results.

Table 5.11: Comparison of the MSE using different boundaries for the soft constraint preferred series length of night shifts varying between 3 and 8 night shifts in a row. The bold number represent the best results per customer case.

Maximum length	**Outlier (1.5*IQR)**	**3**	**4**	**5**	**6**	**7**	**8**
A1	6	**0**	**0**	**0**	**0**	5	17
C1	3	5	5	5	5	**4**	**4**
C2	5	**0**	**0**	**0**	**0**	5	10
C3	5	**0**	**0**	**0**	**0**	2	5
C4	9	**0**	1	1	1	6	17

Summary

- Using the elbow method gives the best results in identifying the preferred series length of night shifts. This is in contrast with the constraint type *preferred series length*, where using the elbow method did not give the best results.
- Only 2 iterations of the algorithm have to be applied.
- Using domain knowledge, we set the upper bound of the preferred series length of night shifts equal to 5 night shifts in a row. By setting the upper bound equal to 5, the MSE reduces with 4 to 17. Therefore, using this domain knowledge improves the performance of the algorithm.

5.4.2 Evaluation of results

In this section, we evaluate the performance of the algorithm we propose to identify the preferred series length of night shifts.

Table 5.12 shows the results of applying the algorithm on customer databases. For customer case A1, C2, and C3, the algorithm could identify the preferred series length of night shifts with an MSE equal to 0, which means that the algorithm could perfectly identify the preferences of the customer. In customer case C4, applying the algorithm results in an MSE equal to 1, meaning that the identified range deviates only 1 from the actual preference.

Table 5.12: Results of identifying the preferred series length of night shifts per customer case

Case	**Identified preference**		**Actual preference**		**Squared error**		**MSE**	**Clusters**
	Minimum	*Maximum*	*Minimum*	*Maximum*	*Minimum*	*Maximum*		
A1	2	3	2	3	0	0	0	2
C1	3	3	1	4	4	1	5	2
C2	2	3	2	3	0	0	0	2
C3	2	4	2	4	0	0	0	2
C4	2	4	2	3	0	1	1	2

However, in case C1 the algorithm performed much worse, resulting in an MSE equal to 5. Figure 5.13 shows the distribution of the length of night shifts series and the results of the K-Means algorithm. The figure shows a peak at a series length equal to three. Consequently, the K-Means algorithm identified only a series length equal to 3 as desirable. Whereas, the customer perceives a series length of night shifts between 1 and 4 as desirable. The algorithm did not identify the preference wrong, as the customer finds a series length of 3 also desirable. However, the actual range of preferred series lengths is wider. This clarifies the higher MSE for this specific customer case.

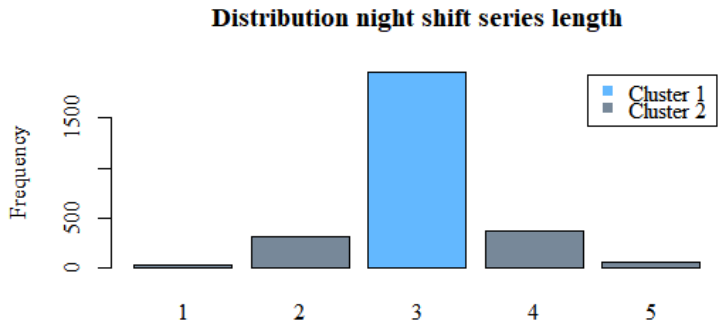


Figure 5.13: Distribution of the length of night shift series and the identified clusters for customer C1

Summary

- In 4 out of 5 customer cases, the algorithm identified the preference of the customer without any error.
- In one of the customer cases, the error was equal to 5. In this case, the algorithm did not identify the preferred series length for night shifts incorrectly, however, the algorithm suggested a smaller range than the customer perceived as desirable.

5.5 Conclusion

The objective of this chapter is to answer the following research question:

For each of the formulated soft constraints, to what extent can the parameters be identified based on historical data?

In general, for all constraints, the methods we propose can distinguish frequent patterns from infrequent patterns. Assuming that these frequent patterns represent scheduling preferences, we can formulate the soft constraints for each customer case. However, in some cases, the preferences as identified by the customer were not in line with the historical schedule. This could mean that the customer did not succeed in defining their real scheduling preferences or that the customers were not able to satisfy these preferences in the past. Since preferences are subjective in nature, it is difficult to precisely define to what extent the methods can identify preferences. The remainder of this section describes our findings for each of the constraints separately.

Consecutive duties

Association rule mining was able to identify 40% to 100% of the soft constraints. Using known hard constraints has a positive effect on the performance of the algorithm. Especially the performance of identifying undesirable combinations of duties improves by applying hard constraints. Furthermore, in general, the algorithm performs better in detecting desirable combinations of duties compared to undesirable combinations. Especially for undesirable combinations of duties, not all identified soft constraints are relevant. In practice, the consultant and planner should pay attention to the following: i) Are all identified soft constraints relevant? and ii) Do we miss

any relevant soft constraints?

Free weekends

Association rule mining was able to identify the preferred percentage of free weekends with a small error between 0.00 and 0.03. Using domain knowledge to select the employees, improves the MAE of the fraction of free weekends for two out of three customer cases. The method is able to identify the preferred fraction of free weekends for employees. However, the algorithm could not identify the length of the period included in the constraint correctly. Therefore, in practice, the consultant and planner should verify whether the period is identified correctly by the algorithm.

Series length

For one customer, the nested K-Means algorithm successfully identified the boundaries for the preferred series length for each type of contract hours with an MSE varying between the 0.16 and 0.90. The preference of the other customer was not in line with the historical schedule, resulting in a higher MSE. In this customer case, using domain knowledge to determine the boundary of the soft constraint, did not result in an improvement of the results. Furthermore, from the data, we cannot derive if the identified series length is a real preference or not. In practice, the consultant and planner should pay attention to the following: i) Does the historical schedule satisfy the preference for the series length? and ii) Do we really have a preference for the series length for each type of contract?

Distribution of night shifts

The K-Means algorithm could identify the preferred series length of night shifts for 4 out of 5 customers cases without an error. Moreover, using domain knowledge has a positive effect on the results as the MSE decreases with at least 2 and at most 9. Additionally, we propose an alternative approach to identify the number of night shifts employees should work. However, this approach uses a different structure of the constraint than currently used in OWS. For that reason, we reflect on this approach in *Chapter 6 Conclusion and Discussion.*

Chapter 6

Conclusion and discussion

In the previous chapters, we answer the sub-questions of this research. In Section 6.1, we bring these conclusions together and provide an answer to the main research question. Section 6.2 provides a discussion of the research, including a description of its scientific and practical contribution. Finally, in Section 6.3 we give recommendations for ORTEC how to use the method we propose and provide ideas for further research.

6.1 Conclusion

Incorporating scheduling preferences in an optimization algorithm is challenging and time-consuming. This research aims to explore how historical shift schedules and domain knowledge can be used to overcome this problem. In this section, we answer our main research question:

How can ORTEC use historical manually-scheduled schedules and domain knowledge to formulate customer-specific soft constraints for the Optimizer?

First, before historical data can be used to detect scheduling preferences, we need to define which soft constraints represent the scheduling preferences of customers. Using data from three customers of ORTEC, we defined the following four categories of soft constraints: i) Desirable and undesirable combinations of consecutive duties, ii) Minimum number of free weekends, iii) Preferred length of a series and iv) Distribution of night shifts. Subsequently, domain knowledge is used to determine the boundaries of these soft constraints. Finally, data mining techniques are used to identify frequent and infrequent patterns in historical data. These patterns can be used to define the soft constraints for the customers. Applying this method on real customer cases gives the following insights:

- For the constraint type, *consecutive duties*, association rule mining could identify 40% to 100% of the most important desirable and undesirable combinations of duties. For *desirable* combinations of duties, 50% to 85% of the identified soft constraints, are indeed a preference of the customer. For *undesirable* combinations of duties, this percentage is lower, only 10% to 35% of the identified soft constraints are indeed a disfavor of the customer. Using the domain knowledge gives insights in which combinations of duties are not possible in

practice. As a result, less irrelevant soft constraints are being identified, which improves the precision of the algorithm with 18% to 533%.

- For all customer cases, association rule mining can identify the fraction of *free weekends* an employee should at least have with a very small error. However, the algorithm could not identify the length of the period the constraint holds for. Using domain knowledge reduces the MAE in two out of three customer cases.

- Using a nested clustering technique, we could identify the range of series length that occurred frequently in the past per employee group. For customer C, the MSE lies between 0.07 and 0.90, indicating good performance of the algorithm, However, for customer B, the MSE is much higher and lies between the 1.68 and 4.55. For this customer, the historical shift schedules are not in line with the preferences of the customer. For this constraint type, adding domain knowledge to determine the boundary of the soft constraint did not improve the performance of the algorithm.

- Using a clustering technique, we could identify the preferred series length of a night shift series. For 4 out of 5 customers, we could identify the preference with an MSE between 0.0 and 1.0, which indicates a good performance of the algorithm. Using domain knowledge to determine the boundary of the soft constraint, improves the performance of the algorithm with 94% to 100%.

- Currently, the constraint type *number of night shifts* is defined on department level. However, data and domain knowledge both indicate that the preferred number of night shifts an employee should work is specific for each employee. For that reason, we propose a method which identifies the preferred number of night shifts on *group level* rather than *department level*. Since we propose a different structure of the constraint, we could not verify the results of this approach. In Section 6.3 we elaborate further on how ORTEC should use our approach and provide ideas for further research.

In conclusion, the methods we propose can identify frequent patterns in historical schedules. In this research, most customers perceived these frequent patterns as preferred and we could identify the soft constraints with only a small error. However, in some cases, the historical shift schedule was not in line with the current preferences. This could mean that the customer did not succeed in formulating the soft constraints correctly, or could not satisfy their preferences in the past. Therefore, in practice, it is important to verify if the historical schedule is preferred. Furthermore, the algorithms succeeded in identifying the parameters of the soft constraints, however, it could not detect if a soft constraint is indeed a preference. Consequently, human-interaction is required to verify if the preference exists or not. We conclude that ORTEC can use historical schedules and domain knowledge to formulate soft constraints for the Optimizer, however, human-interaction is required to fine-tune and verify these soft constraints.

6.2 Discussion

Scientific contribution

With this research, we contribute to the existing literature in the following ways:

- In the literature, several methods exist to learn constraints from historical data. Some of these methods focus on learning hard constraints (Beldiceanu and Simonis, 2012) (Kumar et al., 2018). Other methods focus on learning the weights of soft constraints (Rossi and Sperduti, 2004) (Teso et al., 2016). Relatively few approaches exist to learn the parameters of soft constraints. Moreover, existing methods focused either on small problems (Campigotto et al., 2015) or artificially created data (Hassan, 2019). In contrast to existing literature, this thesis focuses on learning the parameters of soft constraints in complex real-world scheduling problems. We show how data mining can be used to learn the parameters of soft constraints in complex real-world scheduling problems, which is a new field in operations research.
- Furthermore, in this thesis, we use domain knowledge to determine the boundaries of soft constraints. Using domain knowledge in constraint learning was already suggested by Campigotto et al. (2011), however, in the literature this, was not applied in constraint learning. This thesis shows that adding domain knowledge increases the performance of the algorithms and is a valuable addition in learning constraints.
- We have the following two recommendation when using data techniques to identify preferences:
  - In data mining, two types of techniques exist: i) Supervised learning and ii) Unsupervised learning. Supervised learning requires training data including known parameters and uses this to identify parameters in new data. Since preferences are organization or department specific, we do not know preferences for new customers. For that reason, possible methods are limited to unsupervised learning.
  - Furthermore, it is important to keep in mind that each customer has a unique distribution of the data which is not known on forehand. We can distinguish parametric and non-parametric learning models. Parametric learning models use a function of a specific form that needs to match with the data. While non-parametric learning models require the user to make less prior assumptions. Therefore, we recommend using non-parametric learning models.

Practical contribution

As described in Chapter 6, incorporating scheduling preferences in the Optimizer is difficult for several reasons. Among other reasons, planners find it hard to quantify their preferences and preferences are customer-specific. Our method supports the planner and consultant in this process. The method gives insights into what happened in the past and quantifies the preferences of the customers. Our method uses customer specific historical schedules, and thus provides customer specific insight about scheduling preferences. By doing so, we deal with the

fact that preferences are personal and vary within and between organizations. The information provided could be used as a handout during the implementation of the Optimizer. We expect that this saves time and effort in incorporating scheduling preferences in the Optimizer for both the planner and the consultant. In this way, more customers could implement the Optimizer in a shorter period.

Nowadays, some industries have to deal with a tense labor market. This means that organizations are experiencing difficulties to find qualified personnel. Especially in health care organizations, this problem is expected to grow in the future. If we look to the research from a broader perspective, the results contribute to the transformation organizations have to make to use operations research to solve their shift scheduling problem. As described before, operations research has the potential to create efficient shift-schedules and improve employee satisfaction. By doing so, we expect that organizations can cope in a better way with the tense labor market and keep employees satisfied.

Limitations

This research has the following limitations:

- The method we propose is based on the assumption that historical shift schedules are perceived as desirable and contain implicit preferences. The results of this thesis show that in most customer cases, the historical schedules are in line with the current scheduling preferences of the customers. However, in some cases, the scheduling preferences were not in line with what happened in the past. This could mean that the customer did not succeed in identifying the preferences of the customer, or that the historical shift schedules were not desirable. In conclusion, it is important to verify if a customer find its historical shift schedule desirable. If not, the method we propose will suggest soft constraints that are not in line with the preferences of the customer.

- In this research, four categories of soft constraints represent the scheduling preferences of the customers. We defined these categories based on three different customer cases. Two of these customers operate in health care and one customer operates in retail. Moreover, these customers operate in two different countries: the Netherlands and Finland. By including customer cases operating in different industries and different countries, we hope that the types of soft constraints are transferable to other customers. However, we do not know if the four categories of soft constraints fully cover the scheduling preferences of other organizations. Especially in other countries and other industries, the scheduling preferences may be different.

- Most of the algorithms used in this thesis require some additional input parameters, such as thresholds. The results show it is not possible to determine one uniform threshold that results in optimal results for all customer cases. For new customers, we do not know the distribution of the data. As a result, it is not possible to define an optimal customer-specific threshold. For this reason, the performance of the algorithm varies between customer cases.

6.3 Recommendation and future research

Recommendation for ORTEC

We have the following recommendations for ORTEC on how to use the results of this thesis:

- The method we propose uses historical shift schedules of customers. Since ORTEC does not have access to historical data of new customers, this method cannot be applied to new customers. However, currently, a vast amount of customers does not use the optimizer. This means that historical data is available for these customers. Since the Optimizer is used limited, there is still a lot of space for improvement at the current customer base. We recommend ORTEC to use the method we propose to implement the Optimizer at their existing customers.
- Furthermore, we recommend ORTEC to use the output of the algorithms as a handout during the implementation of the soft constraints in the Optimizer. The results quantify the soft constraints for the optimization problem. These soft constraints can be used as a first draft. However, since the algorithm is not able to identify if the soft constraints is indeed a preference or not, the consultant should verify with the customer if the preference exists or not. Furthermore, the consultant should verify if the historical shift schedules are preferred and, if required, fine-tune the parameters of the soft constraints.
- Additionally, in the current method, general domain knowledge is used to define the boundaries of the soft constraints. We show that this improves the performance of the algorithms and results in a better estimation of the scheduling preferences of the customers. We recommend ORTEC to adapt the domain knowledge used to customer specific situations. In this way, ORTEC can define the boundaries of the soft constraints more specifically per customer case. We expect that this results in a better estimation of the scheduling preferences.

Future research

We suggest the following topics for future research:

- **Learning weights**
  In this research, the focus lies on identifying the parameters of soft constraints. The next step in the implementation process of the soft constraints is defining the weights. We recommend ORTEC to investigate if historical data can be used to define the weights of the soft constraints. Besides, if the weights of the soft constraints are being learned, we can distinguish preferences from non-preferences. Constraints that are not a real scheduling preference, should have a weight equal or close to zero. The literature suggests several methods to learn the weights of the soft constraints (Rossi and Sperduti, 2004) (Teso et al., 2016) (Campigotto et al., 2015). We recommend ORTEC to investigate if one of these methods can be used to learn the relative weights of the identified soft constraints.

- **Individual level**

  In this research, we identify scheduling preferences on department level. However, since preferences are personal, scheduling preferences may differ between the employees. The methods we propose can be used on individual level, however, we should keep in mind that the amount of data is limited on individual level. Additionally, different types of soft constraints may represent the preferences of individual employees. Therefore, we recommend ORTEC to investigate how individual scheduling preferences can be incorporated in the Optimizer. We recommend to start with qualitative research, to determine what preferences typically exists at employees. Thereafter, historical shift schedules can be used to identify these preferences on individual level.

- **Incorporate time**

  As stated in Chapter 1, scheduling preferences may change over time. The method we propose is based on historical shift schedules and provides a first draft of the soft constraints for the Optimizer. However, once the Optimizer has been implemented, the soft constraints can get outdated if the preferences of the organization or employees change. We recommend investigating how to keep soft constraints updated. An idea is to monitor the changes a planner makes after the Optimizer created a shift schedule and the change requests from customers. By monitoring these changes, ORTEC can identify which scheduling decisions made by the Optimizer are not preferred. In this way, the soft constraints could be updated continuously without the intervention of the consultant or the customer.

- **Distribution of night shifts**

  Since working night shifts strongly influences the natural sleep rhythms of employees, stricter labor rules apply. Moreover, employees show an individual preference for working night shifts. Consequently, it difficult to schedule night shifts. We propose a method that identifies the preferred number of night shifts series employees should work. This approach is based on the assumption that three types of employee exist within a department: i) Employee with a preference for working night shifts, ii) Employees with a rejection for working night shifts iii) Employees without a preference or rejection for working night shifts. However, due to a limited amount of data, we could not verify this assumption. We recommend ORTEC to further investigate how night shifts are distributed among the employees at their customers before implementing this approach.

Bibliography

Beldiceanu, N. and Simonis, H. A Model Seeker: Extracting Global Constraint Models from Positive Exampels. In *Principles and Practice of Constraint Programming*, pages 141–157, Québec City, 2012. Springer. doi: 10.1007/978-3-642-33558-7.

Blöchliger, I. Modeling staff scheduling problems. A tutorial. *European Journal of Operational Research*, 158(3):533–542, nov 2004. doi: 10.1016/S0377-2217(03)00387-4.

Brooks, I. and Swailes, S. Analysis of the relationship between nurse influences over flexible working and commitment to nursing. *Journal of Advanced Nursing*, 38(2), 2002. doi: https://doi.org/10.1046/j.1365-2648.2002.02155.x.

Brucker, P., Qu, R., and Burke, E. Personnel scheduling: Models and complexity. *European Journal of Operational Research*, 210(3):467–473, 2011. doi: 10.1016/j.ejor.2010.11.017.

Burke, E. K., Trick De Causmaecker, P. A., Vanden Berghe, G., and Landeghem, H. V. THE STATE OF THE ART OF NURSE ROSTERING. Technical report, 2004.

Campigotto, P., Passerini, A., and Battiti, R. Active Learning of Combinatorial Features for Interactive Optimization. In *Learning and Intelligent Optimization*, pages 336–350, Rome, 2011. Springer. doi: 10.1007/978-3-642-25566-3.

Campigotto, P., Battiti, R., Passerini, A., Lehrstuhl, I., and Universit, T. Learning Modulo Theories for preference elicitation in hybrid domains. *CoRR abs:1508.04261*, 2015.

De Grano, M. L., Medeiros, D. J., and Eitel, D. Accommodating individual preferences in nurse scheduling via auctions and optimization. *Health Care Management Science*, 12(3):228–242, 2009. doi: 10.1007/s10729-008-9087-2.

De Raedt, L., Passerini, A., and Teso, S. Learning Constraints from Examples. pages 7965–7970, 2018.

Drakos, G. How to select the Right Evaluation Metric for Machine Learning Models: Part 1 Regression Metrics, 2018.

Drouin, B. R. and Potter, M. Flexible Scheduling. *The American Journal of Nursing*, 105(11), 2005.

El Adoly, A. A., Gheith, M., and Nashat Fors, M. A new formulation and solution for the nurse scheduling problem: A case study in Egypt. *Alexandria Engineering Journal*, 57(4):2289–2298, dec 2018. doi: 10.1016/J.AEJ.2017.09.007.

Hair, J. and Black, W. *Cluster Analysis*. 2000. doi: 10.1016/B978-012691360-6/50012-4.

Han, J., Kamber, M., and Pei, J. *Introduction*. 2012a. doi: 10.1016/B978-0-12-381479-1.00001-0.

Han, J., Kamber, M., and Pei, J. *Mining Frequent Patterns, Associations, and Correlations*. 2012b. doi: 10.1016/b978-0-12-381479-1.00006-x.

Han, J., Kamber, M., and Pei, J. *Classification*. 2012c. doi: 10.1016/B978-0-12-381479-1.00009-5.

Hassan, S. A. Learning Constraints from Human-Planned Employee Schedules. Technical report, 2019.

Jaumard, B., Semet, F., and Vovor, T. A generalized linear programming model for nurse scheduling. *European Journal of Operational Research*, 107(1):1–18, may 1998. doi: 10.1016/S0377-2217(97)00330-5.

Kellogg, D. L. and Walczak, S. Nurse Scheduling: From Academia to Implementation or Not? *INFORMS Journal on Applied Analytics*, 37(4):355, 2007. doi: 10.1287/inte.1070.0291.

Kotsiantis, S. and Kanellopoulos, D. Association Rules Mining: A Recent Overview. *Science*, 32(1):71–82, 2006.

Kumar, M., Teso, S., and De Raedt, L. Automating Personnel Rostering by Learning Constraints Using Tensors. (1), 2018.

Kwak, S. K. and Kim, J. H. Statistical data preparation: Management of missing values and outliers. *Korean Journal of Anesthesiology*, 70(4):407–411, 2017. doi: 10.4097/kjae.2017.70.4.407.

Lin, W.-Y. and Tseng, M.-C. Automated support specification for efficient mining of interesting association rules. *Journal of Information Science*, 32(3):238–250, 2006. doi: 10.1177/0165551506064364.

Ministerie van Sociale Zaken en Werkgelegenheid. The Working Hours Act. Technical report, 2010.

Ministry of Economic Affairs and Employment of Finland. Working Hours Act. 2017.

Post, G. and Veltman, B. Harmonious personnel scheduling. *Proceedings of the 5th International Conference on the Practice and Theory of Automated Timetabling. PATAT*, pages 557–559, 2004.

Rossi, F. and Sperduti, A. Acquiring Both Constraint and Solution Preferences in Interactive Constraint Systems. Technical report, 2004.

Silvestro, R. and Silvestro, C. An evaluation of nurse rostering practices in the National Health Service. *Journal of Advanced Nursing*, 32(3):525–535, 2000. doi: 10.1046/j.1365-2648.2000.01512.x.

Smet, P., Causmaecker, P. D., Bilgin, B., and Vanden Berghe, G. Nurse Rostering : A Complex Example of Personnel Scheduling with Perspectives Nurse rostering : a complex example of personnel scheduling with perspectives. (March 2018), 2013. doi: 10.1007/978-3-642-39304-4.

Solos, I. P., Tassopoulos, I. X., and Beligiannis, G. N. A generic two-phase stochastic variable neighborhood approach for effectively solving the nurse rostering problem. *Algorithms*, 6(2): 278–308, 2013. doi: 10.3390/a6020278.

Tan, P.-N., Steinbach, M., and Kumar, V. Association Analysis: Basic Concepts and Algorithms. *Introduction to Data mining*, pages 327–414, 2005. doi: 10.1111/j.1600-0765.2011.01426.x.

Teso, S., Passerini, A., and Viappiani, P. Constructive preference elicitation by setwise max-margin learning. In *IJCAI International Joint Conference on Artificial Intelligence*, volume 2016-Janua, pages 2067–2073, 2016.

Van Den Bergh, J., Beliën, J., De Bruecker, P., Demeulemeester, E., and De Boeck, L. Personnel scheduling: A literature review. *European Journal of Operational Research*, 226(3):367–385, 2013. doi: 10.1016/j.ejor.2012.11.029.

Van der Veen, E., Hurink, J., Schutten, J., and Uijland, S. A flexible iterative improvement heuristic to support creation of feasible shift rosters in self-rostering. *Annals of Operations Research*, 239(1):159–206, 2016. doi: https://doi.org/10.1007/s10479-014-1540-7.

Wu, X., Zhang, C., and Zhang, S. Efficient mining of both positive and negative association rules. *ACM Transactions on Information Systems*, 22(3):381–405, 2004. doi: 10.1145/1010614.1010616.

Appendix A

Preferences consecutive duties

Customer A - Department A1

- Consecutive shifts are the same
  Consecutive shifts should have the same duty types
- Combinations of consecutive shifts
  After a shift with [DutyType1] no shift with [DutyType2]
  - A, 01
  - 05, 01
  - 05, A
  - 05V, 011V
  - 05V, D81E
  - 05L, 011L
  - AC, AD
  - AD, AC

Customer C - Department C1

- Change in begin-time of shifts
  A shift on the next day starts maximally [NHours earlier] earlier and [NHours later] later.
  - TimeAfter: 1 hour
  - TimeBefore: 8 hours
- Combinations of consecutive shifts
  After a shift with [DutyType1] no shift with [DutyType2]
  - 1Y, 2Y
  - 2Y, 1Y

Customer C - Department C2

- Change in begin-time of shifts
  A shift on the next day starts maximally [NHours earlier] earlier and [NHours later] later.
  - TimeAfter: 1 hour
  - TimeBefore: 8 hours

Customer C - Department C3

- Consecutive shifts are the same
  Consecutive shifts should have the same duty types
- Combinations of consecutive shifts
  After a shift with [DutyType1] no shift with [DutyType2]
  - Y1, Y2
  - Y1, Y3
  - Y2, Y1
  - Y2, Y3
  - Y3, Y1
  - Y3, Y2
- Change in begin-time of shifts
  A shift on the next day starts maximally [NHours earlier] earlier and [NHours later] later.
  - TimeAfter: 1 hour
  - TimeBefore: 8 hours

Customer C - Department C4

- Combinations of consecutive shifts
  After a shift with [DutyType1] no shift with [DutyType2]
  - 1Y, YOA
  - 1Y, YOY
  - 2Y, YOA
  - 2Y, YOY
- Change in begin-time of shifts
  A shift on the next day starts maximally [NHours earlier] earlier and [NHours later] later.
  - TimeAfter: 1 hour
  - TimeBefore: 8 hours

Appendix B

Preferences number of free weekends

Constraint is formulated as follows:

In a period of [NWeeks] weeks, an employee should have at least [NWeekends] weekends off.

Table B.1: Preferred number of free weekends per period customer D

Department	**Number of free weekends**	**Number of weeks**
B1	2	6
B2	2	6
B3	2	8

Appendix C

Preferences series length

Constraint is formulated as follows:

A series of shifts should have a minimum length of [Minimum length] and a maximum length of [Maximum length]. This constraints holds only for employees with a contract of minimal [Minimum hours] and maximal [Maximum hours] hours.

Table C.1: Preferences shift series length customer D

Department	**Contract hours**		**Shift Types**	**Preferred series length**	
	Minimum	*Maximum*		*Minimum*	*Maximum*
B1	30	48	All	3	6
B2	30	48	All	3	7
B3	30	48	All	3	7

Table C.2: Preferences shift series length customer C

Department	**Contract hours**		**Shift Types**	**Preferred series length**	
	Minimum	*Maximum*		*Minimum*	*Maximum*
C1	18	48	All	2	5
C2	18	48	All	2	5
C3	31	48	All	2	6
	18	31	All	2	5
C4	31	48	All	2	6
	18	31	All	2	5

Appendix D

Preferences distribution of night shifts

Preferred length of a night shift series

A series of night shifts should have a minimum length of [Minimum length] and a maximum length of [Maximum length].

Table D.1: Preferences shift series length night shifts per customer case

Department	**Shift Types**	**Preferred series length**	
	Minimum	*Maximum*	
A1	Night	2	3
C1	Night	1	4
C2	Night	2	3
C3	Night	2	4
C4	Night	2	3

Number of night shifts

Employees should work between [Minimum] and [Maximum] number of night shifts in a period of [NWeeks] weeks.

Table D.2: Preferences of the number of night shift per customer case

Department	**Period in weeks**	**Preferred number of shifts**	
		Minimum	*Maximum*
A1	4	1	3
C1	3	3	4
C2	3	1	4
C3	3	2	3
C4	3	2	3