Workforce scheduling with logical constraints: theory and applications in call centers

Gábor Danó

This thesis was supervised by
Sandjai Bhulai
and
Ger Koole

Department of Mathematics
Vrije Universiteit
Amsterdam
July 2004

Contents

Introduction

Nowadays, enterprises and organizations are faced with increasing demand requirements. On the one hand, it means that they have to provide high quality of service for their customers. On the other hand, they would like to satisfy their employees' demand as much as possible. Thus, companies such as airlines, hospitals, police and ambulance departments, telephone companies and banks face a very delicate problem with workforce scheduling.
Generally, workforce, labor or personnel scheduling or rostering is the process of designing work timetables for employees to satisfy the demand requirements for its services. Different kind of mathematical approaches have developed in order to help the companies to solve this problem. According to [6] the development of these mathematical models and algorithms involves the following levels:

1. a demand modeling study that collects and uses historical data to forecast demand for services and converts these to the staffing levels needed to satisfy service standards,

2. consideration of the solution techniques required for a personnel scheduling tool that satisfies the constraints arising from workplace regulations while best meeting a range of objectives including coverage of staff demand, minimum cost and maximum employee satisfaction,

3. specification of a reporting tool that displays solutions and provides performance reports.

In this master thesis we focus on problems in the second level, i.e., the shift scheduling and the staff assignment problem. The general shift scheduling problem involves determining the number of employees to be assigned to each shift to satisfy the demand requirements. In the staff assignment problem we have the number of required employees for each shift and we have to assign employees for shifts. It is very important, but meanwhile very difficult to find an optimal solution that minimizes the cost and satisfies employee preferences of these complex and highly constrained problems.

The main interest of this thesis is to introduce and apply a mathematical programming model, which can solve both the shift scheduling and the staff assignment problem. In this model we use logical constraints to model different conditions and requirements by the government or by the company.

In Chapter 1 we give an introduction of the shift scheduling and the staff assignment problems. We discuss different mathematical programming models from the literature, compare them and introduce some basic definitions and terms, which we use afterwards in the thesis.

In Chapter 2 we build up our labor scheduling model. In the beginning of this chapter we introduce the way we can use decision variables in a linear programming (LP) or integer programming (IP) environment to represent connections between certain states of variables. Later in this chapter we give an introduction of mathematical logic and we describe how we can model logical relationships in an LP or IP model. Using these modeling techniques we build up a labor scheduling model, in which we represent shifts implicitly, i.e., we use logical constraints to express them. This model is also described in this chapter.

In Chapter 3 we give an overview of call centers, in which labor scheduling plays a very important role. Moreover, in this chapter we give a basic introduction on queueing theory and queueing models, which provide the mathematical background to model and analyze these centers. We focus on mathematical methods, which we can use to convert historical data to estimate the number of required agents.

In Chapter 4 we solve a workforce scheduling problem with our model. Afterwards in this chapter we describe our modeling and implementation method. In the end of the chapter we analyze the experimental results of the test problems.

In Chapter 5 we make some concluding remarks about our modeling method, computational results and we will mention some future prospects.

I would like to thank my supervisors Sandjai Bhulai and Ger Koole for their attention and support during this semester, which enriched my knowledge and experience in mathematics.

Chapter 1

Models from the literature

In this chapter we would like to introduce the labor scheduling problem. As we already mentioned, it has different steps and for each step there are different mathematical programming models to solve it. In the beginning of this chapter we will deal with three different mathematical programming formulations of the shift scheduling problem. We will describe and analyze these models and we will make a comparison between them. After, we will introduce one mathematical programming model for assigning employees to shifts. All of these models are from the literature.

For dealing with these models it is essential to introduce the following basic definitions and terms, which we will use afterwards in this thesis.

1.1 Definitions of terms and sets

Definition 1.1.1 *A period is a basic interval for which the planning occurs, typically 15 min., 30 min. or 60 min.*

Definition 1.1.2 *A shift is a work schedule for an employee for a specific day, that satisfies different restrictions defined by management policy, contractual obligations or governmental regulations.*

Definition 1.1.3 *Work span is the contiguous set of planning periods in which either work or rest must take place for a given shift.*

Definition 1.1.4 *A tour is a work schedule for an employee for a week, comprised of separate shifts, that satisfies applicable restrictions defined by management policy, contractual obligations or governmental regulations.*

Definition 1.1.5 *A service level is a predefined property of the service for measuring its quality.*

Definition 1.1.6 *A break window is a set of contiguous periods in which the break occurs.*

Definition 1.1.7 *The extraordinary overlap is the existence of the following condition: the initial and final periods in the break window for any one shift type are earlier and later respectively, than the corresponding initial and final periods in the break window for at least one other shift type.*

1.2 Model of Dantzig

The shift scheduling problem was introduced first by Edie [5] in 1954. Since his paper this problem and a number of related ones have attracted considerable attention in the literature. The first integer programming formulation for the shift scheduling problem presented by Dantzig [4] is the following set-covering type model (DLSM):

$$\min z = \sum_{t \in T} c_t x_t, \tag{1.1}$$

subject to

$$\sum_{t \in T} a_{tp} x_t \geq r_p \qquad \text{for} \quad p \in P, \tag{1.2}$$

$$x_t \in \mathbb{N} \qquad \text{for} \quad t \in T, \tag{1.3}$$

where the sets and variables are the following:

P the set of indices associated with planning periods,
T the set of indices associated with allowable shifts (tours),
a_{tp} $\begin{cases} 1, & \text{if period } p \text{ is a working period for shift (tour) } t, \\ 0, & \text{otherwise,} \end{cases}$
c_t the cost of having an employee work shift (tour) t,
r_p the desired staffing level in period p,
x_t the number of employees working shift (tour) t.

In this DLSM model the objective (1.1) is to minimize the cost of the scheduled shifts (tours), subject to restrictions (1.2) and (1.3).
Constraint set (1.2) ensures that sufficient employees are present in all periods.
If we give different shift types, lengths, shift start times, number and length of relief (rest), lunch breaks and break windows, the set-covering formulation requires that all possible combinations of these features are included as different shifts in T.
When relief and lunch breaks and break windows are included, the number of shifts in T increases rapidly. Consequently, standard integer programming tools fail to be useful to solve these extended models because of the extremely high number of shifts in T.

1.3 Model of Bechtold and Jacobs

Bechtold and Jacobs [3] in 1990 applied implicitly matched breaks with explicitly modeled shifts. They began to develop this model with the following initial assumptions:

(1) The organization operates less than 24 hours.

(2) All planning periods are of equal length.

(3) Every employee will receive exactly one break.

(4) The duration of the breaks is identical for all shifts that are modeled.

(5) The duration of breaks contains one or more planning periods.

(6) Each shift type is defined by the assignment of a single break window to an associated work span.

(7) The break window for each shift consists of any selected nonzero subset of periods subject to the restriction that all periods contained within the break interval must be a subset of the work span associated with the shift.

(8) Extraordinary overlap does not exist.

(9) No understaffing is allowed.

With these assumptions, their (BJSSM) model is the following:

$$\min \sum_{t \in T} c_t x_t, \tag{1.4}$$

subject to

$$\sum X_p - \sum \Omega_p \geq r_p \quad \text{for } p \in P, \tag{1.5}$$

$$\sum F_k - \sum G_k \geq 0 \quad \text{for all but the last period} \quad k \in N, \tag{1.6}$$

$$\sum R_k - \sum S_k \geq 0 \quad \text{for all but the first period} \quad k \in M, \tag{1.7}$$

$$\sum X - \sum B = 0, \tag{1.8}$$

$$x_t \in X, \;\; x_t \in \mathbb{N}, \tag{1.9}$$

$$b_p \in B, \;\; b_p \in \mathbb{N}, \tag{1.10}$$

where the sets and the variables of the model are the following:

$\sum A$	the sum of all elements contained in any set A,
P	the set of indices associated with planning periods,
T	the set of indices associated with allowable shifts,
X	$\{ x_t \| t \in T \}$,
N	the set of final (latest) periods (ascending order) in the break windows associated with all shifts,
M	the set of initial (earliest) periods (ascending order) in the break windows associated with all shifts,
Ω_p	the set of break variables for which period p is a break period,
c_t	the cost of having an employee work shift t,
r_p	the labor requirement in period p,
x_t	the number of employees working shift t,
b_p	the total number of breaks initiated at the start of period p by the complete set of employees from all shifts,
l	the earliest period in P in which the break for any shift may begin,
u	the latest period in P in which the break for any shift may begin,
W_t	the work span associated with shift t,
X_p	$\{ x_t \| p \in W_t \}$, i.e., the total number of employees working in period p,
V_t	the set of periods associated with the break window for shift t,
U_t	$\{ b_p \| p \in V_t \}$,
F_k	$\{ b_p \| k \in N \text{ and } p = l, l+1, \ldots, k \}$,
G_k	$\{ x_t \| U_t \subseteq F_k \}$, i.e., the set of x_t where t belongs to those shifts for which break windows are in the interval $(l, l+1, \ldots, k)$ with $k \in N$,
R_k	$\{b_p \| k \in M \text{ and } p = k, k+1, \ldots, u \}$,
S_k	$\{ x_t \|U_t \subseteq R_k \}$, i.e., the set of x_t where t belongs to those shifts for which break windows are in the interval $(k, k+1, \ldots, u)$ with $k \in N$,
P_b	$\{ l, l+1, \ldots, u-1, u \}$,
B	$\{ b_p \| p \in P_b \}$, i.e., the total number of breaks initiated in the interval $\{ l, l+1, \ldots, u-1, u \}$.

In this model the objective (1.4) is to minimize the cost of the scheduled shifts, subject to restrictions (1.5)-(1.10). Constraint set (1.5) ensures that demand requirements are met in all periods. Constraint sets (1.6) - (1.8) are the control constraints, which are included to ensure that the shifts are implicitly represented. Constraint set (1.6) ensures that sufficient quantities of breaks are appropriately available for allocation to subset of shifts with break windows which are fully contained within the successively larger intervals $(l, l+1, \ldots, k)$ with $k \in N$. The function of constraint set (1.7) is the same as constraint set (1.6) except that the successively larger intervals are $(k, k+1, \ldots, u)$ with $k \in M$. Constraint (1.8) ensures that exactly one break is available for each employee scheduled.

The proof that the DLSM model is equivalent with this BJSSM is contained in [2] which was based upon the initial assumptions. The term equivalence is used to indicate that the two models represent the same set of allowed shifts and that the optimal objective function values for the two models are equal.

1.4 Model of Aykin

Aykin's model [1] is also based on the implicit representation of the break placements. He considered the problem with multiple rest and lunch breaks and multiple break windows. He introduced integer variables for the numbers of employees assigned to a shift and starting their breaks in different planning periods within the associated break window. This representation has high flexibility provided by the break windows, i.e., the number and the length of the break windows are optional in this model.
Thus, without loss of generality he assumed that every employee receives two relief breaks and one lunch break. Further, it was also assumed that the duration of the relief breaks is one period and the duration of the lunch break is two periods.
With these assumptions, Aykin's general shift scheduling model (ASSM) can be formulated as follows:

$$\min \sum_{t \in T} c_t x_t, \tag{1.11}$$

subject to

$$\begin{aligned} \sum_{t \in T} a_{tp} x_t - \sum_{t \in T1_p} u_{tp} - \sum_{t \in TL_{(p-1)}} w_{t(p-1)} - \\ - \sum_{t \in TL_p} w_{tp} - \sum_{t \in T2_p} v_{tp} \quad &\geq \quad b_p \quad \text{for } p \in P, &(1.12)\\ x_t - \sum_{p \in B1_t} u_{tp} \quad &= \quad 0 \quad \text{for } t \in T, &(1.13)\\ x_t - \sum_{p \in BL_t} w_{tp} \quad &= \quad 0 \quad \text{for } t \in T, &(1.14)\\ x_t - \sum_{p \in B2_t} v_{tp} \quad &= \quad 0 \quad \text{for } t \in T, &(1.15)\end{aligned}$$

$$x_t,\, u_{tp},\, w_{tp},\, v_{tp} \in \mathbb{N}, \tag{1.16}$$

where the sets and the variables in the model are the following:

x_t the number of employees working shift t,
b_p the number of employees needed in period p,

a_{tp} $\begin{cases} 1, & \text{if period } p \text{ is a working period for shift (tour) } t, \\ 0, & \text{otherwise,} \end{cases}$

u_{tp} the number of employees assigned to shift t and starting their first relief break in period p,

w_{tp} the number of employees assigned to shift t and starting their lunch break in period p,
v_{tp} the number of employees assigned to shift t and starting their second relief break in period p,
T the set of indices associated with allowable shifts,
P the set of indices associated with periods,
$B1_t$ the set of planning periods in which an employee working shift t may start his/her first relief break,
$B2_t$ the set of planning periods in which an employee working shift t may start his/her second relief break,
BL_t the set of planning periods in which an employee working shift t may start his/her lunch break,
$T1_p$ the set of shifts for which period p is a break start time within the time windows for the first relief break,
$T2_p$ the set of shifts for which period p is a break start time within the time windows for the second relief break,
TL_p the set of shifts for which period p is a break start time within the time windows for the lunch break.

In this scheduling model the objective (1.11) is to minimize the cost of the scheduled shifts subject to the constraint sets (1.12)-(1.16). With constraint set (1.12) we ensure that sufficient number of employees are present in all periods. Every employee has to take exactly one first, one second relief break and one lunch break. We model these conditions with constraint sets (1.13)-(1.15), where constraint set (1.13) is responsible for the first relief breaks, constraint set (1.14) is responsible for the lunch breaks and constraint set (1.15) is responsible for the second relief breaks.

This model can be extended with additional restrictions on break placements and shift types.
For example, Aykin suggested the following constraint to model the situation when the number of employees taking a break in a period p has to be limited because of space availability or company policy.

$$\sum_{t \in T1_p} u_{tp} + \sum_{t \in TL_{(p-1)}} w_{t(p-1)} + \sum_{t \in TL_p} w_{tp} + \sum_{t \in T2_p} v_{tp} \leq h_p. \tag{1.17}$$

It is possible to model shifts with different lengths or number of breaks with this formulation by introducing a new set of break variables and new break constraints for every additional break.

1.5 Comparison of the models

In the previous sections we introduced three different formulations of the shift scheduling problem: Dantzig [4], Bechtold and Jacobs [3] and Aykin [1].

In this section we compare these models, focusing on the number of variables and the number of constraints in them.

First, we will introduce notations for the cardinality of the following sets:

$$\begin{array}{lcll}
|P| & = & n_P, & \text{where } P \text{ is the set of indices associated with planning periods,} \\
|T| & = & n_T, & \text{where } T \text{ is the set of indices associated with allowable shifts,} \\
|N| & = & n_N, & \text{where } N \text{ is the set of final (latest) periods (ascending order)} \\
 & & & \text{in the break windows associated with all shifts,} \\
|M| & = & n_M, & \text{where } M \text{ is the set of initial (earliest) periods (ascending order)} \\
 & & & \text{in the break windows associated with all shifts,} \\
|V_t| & = & n_{V_t}, & \text{where } V_t \text{ is the set of periods associated} \\
 & & & \text{with the break window for shift } t, \\
|B| & = & n_B, & \text{where } B = \{\, b_p | \, p \text{ is in the set of break periods} \} \text{ and } b_p \\
 & & & \text{is the total number of breaks initiated at the start of period } p, \\
|B1_t| & = & n_{1t}, & \text{where } B1_t \text{ is the set of planning periods in which an employee} \\
 & & & \text{working shift } t \text{ may start his/her first relief break,} \\
|BL_t| & = & n_{Lt}, & \text{where } BL_t \text{ is the set of planning periods in which an employee} \\
 & & & \text{working shift } t \text{ may start his/her lunch break,} \\
|B2_t| & = & n_{2t}, & \text{where } B2_t \text{ is the set of planning periods in which an employee} \\
 & & & \text{working shift } t \text{ may start his/her second relief break.}
\end{array}$$

Let us denote with $NC(P)$ the number of constraints in any mathematical programming model P and with $NV(P)$ the number of variables in that model.
In our analysis, we deal with the DLSM, the BJSSM and the ASSM scheduling models and analyze them from this point of view .

With the introduced notations, we have the following results:

$$NC(DLSM) = n_P, \tag{1.18}$$

$$NV(DLSM) = \sum_{t \in T} n_{V_t} = \sum_{t \in T} n_{1t} n_{Lt} n_{2t}, \tag{1.19}$$

$$NC(BJSSM) = n_P + n_M + n_N - 1, \tag{1.20}$$

$$NV(BJSSM) = n_T + n_B, \tag{1.21}$$

$$NC(ASSM) = 3n_T + n_P, \tag{1.22}$$

$$NV(ASSM) = n_T + \sum_{t \in T} \big(n_{1t} + n_{Lt} + n_{2t}\big). \tag{1.23}$$

Equation (1.18) comes directly from the constraint set (1.2) in the model of Dantzig on page 6. In that model we formulate all shifts implicitly; thus on one hand, if we have

V_t break periods for any $t \in T$ then we have altogether $\sum_t n_{V_t}$ variables in it. On the other hand, if we have n_{1t}, n_{2t}, n_{Lt} number of periods for the first, the second relief break and the lunch break respectively for any $t \in T$, we should multiply them and sum the result over all shifts t. This result is shown in equation (1.19).

In the model of Bechtold and Jacobs on page 7, we have the constraint set (1.6) for all periods in N except the latest period and the constraint set (1.7) for all periods in M except the earliest one. This implies that we have $n_N - 1 + n_M - 1$ constraints from these sets, n_P from the constraints in equation (1.5) and an additional one from equation (1.6). Hence, we get the result in (1.20). Result (1.21) is derived from constraints (1.9) and (1.10) on page 7.

In Aykin's model on page 9, we assumed that there are three breaks for every employee. This assumption has a big effect on the number of constraints and variables in formulas (1.22) and (1.23). The previous result comes from constraint set (1.12), where we have n_P constraints and from equation (1.12)-(1.14), where we have n_T constraints for each break. In this model we have one variable for each shift t and n_{1t}, n_{2t} and n_{Lt} for any period t. These results lead us to equation (1.23).

We can easily make a comparison between the DLSM and the BJSSM model. From result (1.18) and (1.20) is it clear that in the model of Dantzig we have less constraints. If we would like to compare the number of variables in these two models, we have to do a more in-depth analysis. If break flexibility is not modeled, then $NV(BJSSM) = n_T$, so we have less variables in the DLSM model. However, the relative advantage in using the BJSSM formulation increases significantly when n_{V_t} is increasing, because in this case the number of variables increases rapidly in the Dantzig model.

In the comparison between Dantzig's model and the model of Aykin we can see that there are less constraints than in the previous model, i.e., if we compare equation (1.18) and (1.22). As for the variables, in the Dantzig model we have a product of three variables (or even more if we have more breaks) and in the Aykin model we have the sum of them. From this we can conclude that if n_{1t}, n_{2t} and n_{Lt} are big enough then Aykin's model has less variables.

If we would like to compare the BJSSM and the ASSM model we have a little problem, because these models model different scheduling problems. The main difference between them is the number and the length of breaks: we have three breaks in Aykin's model (one is two periods) and one break in Bechtold and Jacobs model. To be able to compare them we have to model the similar problem with these formulations.
Fortunately, it is possible to model a scheduling problem with one break, which is one period long, with both models. In this case results of the BJSSM model remain true,

i.e., equation (1.20) and (1.21), but we have to modify Aykin's model in the following way: we have to use only the variables and the constraints, which are for the first relief break. In this case we have n_P constraints from constraint set (1.12) and n_T from constraint set (1.12), which means that we have $n_P + n_T$ constraints in this model, i.e., in this case $NC(ASSM) = n_P + n_T$. This result is comparable with the result in equation (1.20). Using the same logic as above we have that in Aykin's model there are $n_T + \sum_{t \in T} n_{1t}$ variables, which is comparable also with the number of variables in the BJSSM model in equation (1.21).

So far in this chapter we introduced and analyzed three different mathematical programming approaches for the shift scheduling problem. In the next section we will introduce a model for assigning employees to shifts.

1.6 Model of Thompson

In this section we deal with Thompson's model [8] for assigning employees to shifts with some special characteristics. There are some initial restrictions in the model, which are the following:

(1) All shifts have to be assigned. These shifts were identified by the scheduler previously.

(2) It is obliged contractually to satisfy preferences for shifts in order of employee seniority, i.e., a senior employee receives his/her desired maximum number of shifts before a less senior employee receives more than one shift.

(3) Employees have special characteristics:

   (a) availability, i.e., the time when they are available and the maximum number of shifts they can work,
   (b) skills and location,
   (c) all employees receive at least one shift per week,
   (d) an employee cannot be assigned more than one shift per day.

With these restrictions, Thompson's model is the following:

$$\min P_0 \sum_{t \in T} u_t + \sum_{e \in E} P_e \left(\sum_{\{t \in T | c_t \in C_e, d_t \in D_e\}} v_{et} x_{et} \right) \tag{1.24}$$

subject to

$$\sum_{\{e \in E | c_t \in C_e, d_t \in D_e\}} x_{et} + u_t = 1 \quad \text{for } t \in T, \tag{1.25}$$

$$\sum_{\{t \in T | c_t \in C_e, d_t \in D_e\}} x_{et} \geq 1 \quad \text{for } e \in E, \tag{1.26}$$

$$\sum_{\{t \in T | c_t \in C_e, d_t \in D_e\}} x_{et} \leq m_e \quad \text{for } e \in E, \tag{1.27}$$

$$\sum_{\{t \in T | c_t \in C_e, d_t = i\}} x_{et} \leq 1 \quad \text{for } e \in E, \ i \in D_e, \tag{1.28}$$

$$\sum_{\{t \in T | c_t \in C_e, d_t \in D_e\}} x_{et} \leq 1 + (m_e - 1) y_{e-1} \quad \text{for } \{e \in E | e > 1\}, \tag{1.29}$$

$$\sum_{\{t \in T | c_t \in C_e, d_t \in D_e\}} x_{et} \geq m_e y_e \quad \text{for } e \in \{E | e < E\}, \tag{1.30}$$

$$x_{et} \in \{0, 1\} \quad \text{for} \quad e \in E, \ \{t \in T | c_t \in C_e, d_t \in D_e\}, \tag{1.31}$$

$$y_e \in \{0, 1\} \quad \text{for} \quad \{e \in E | e < E\}, \tag{1.32}$$

$$u_t \in \{0, 1\} \quad \text{for} \quad t \in T, \tag{1.33}$$

where the sets and the variables of the model are the following:

T the set of shifts,
E the set of employees, ordered from most senior to least senior,
P_i the priority of component i, where $P_i \gg P_{i+1}$
(the priority for component i is much larger than
the priority for component $i + 1$),
C_e the set of shift categories that employee e can work,
D_e the set of days that employee e can work,

x_{et} $\begin{cases} 1, & \text{if employee } e \text{ is assigned to shift } s, \\ 0, & \text{otherwise,} \end{cases}$

m_e the maximum number of shifts to assign to employee e,
v_{et} the undesirability of assigning employee e to shift t,

y_e $\begin{cases} 1, & \text{if employee } e \text{ is assigned the maximum number of shifts,} \\ 0, & \text{otherwise,} \end{cases}$

c_t the category of shift t,
d_t the day on which shift t is scheduled,

u_t $\begin{cases} 1, & \text{if shift } t \text{ is unassigned,} \\ 0, & \text{otherwise.} \end{cases}$

In this model the objective (1.24) is to minimize the number of unassigned shifts and specify that employees' shift choices are to be satisfied, if possible, in order of seniority.

With constraint set (1.25) we ensure that all shift will be assigned, i.e., minimizing the number of unassigned shifts u_t. We have to assign at least one shift to each employee. We model this condition with constraint set (1.26). Constraint set (1.27) ensures that we assign no more than the maximum desired number of shifts for each employee and constraint set (1.28) ensures that no more than one shift per day per employee is assigned. With constraint set (1.29)-(1.31) we force a more senior employee to work their desired number of shifts before a less senior employee works more than one shift.

In the next chapter we will introduce our workforce scheduling model. In this model we will present and use a different approach for formulating labor scheduling problems.

Chapter 2

Scheduling model with logical constraints

In the previous chapter we introduced three mathematical programming models of the shift scheduling problem and one mathematical programming model for the employee assignment problem. In these shift scheduling models we saw different techniques for modeling the breaks: explicit modeled breaks in Dantzig model [4], implicit modeled breaks in Bechtold and Jacobs's [3] and Aykin's model [1]. In all of these models the shifts are represented by the set T, which means that in these models the allowable shifts are explicitly in the model. In Thompson's model [8] we saw a modeling technique, with which we can assign shifts to employees and satisfy employees' preferences for shifts in order of seniority.

In this chapter we will introduce our labor scheduling model, in which we will model shifts implicitly. For this reason at the beginning of the chapter we will deal with the relationship between mathematical logic and operation research, i.e., the way we can represent logical constraints with LP or IP formulation. Using this formulation we can provide high flexibility with involve different rules to the model, with which we can model shifts in an implicit way. In our model we will distinguish between employees with a long term contract and employees, who can work only in specific periods a day (e.g., students, part time workers, etc.). It is possible that we have employees, who have long term contract, with different skills, experiences and different preferences, i.e., we have different employees for different types of work and they would like to work in different periods (e.g., an employee does not want to work in the night or he/she would like to finish his/her work earlier). We will model these preferences in the following way: we will penalize if an employee has to work in the period, in which he/she does not want and vice versa.

At the end of the chapter we will present our shift scheduling model, which we will use to solve a scheduling problem from call centers in Chapter 4.

2.1 Decision, indicator variables

In a linear programming (LP) or integer programming (IP) environment we may use additional variables to represent connections between different variables and states.
In this section we will introduce the connection between certain states of a continuous variable and states of its indicator or decision variable. In our representation the indicator and decision variables are always 0-1 variables. We will denote the LP or IP variables with x or x_i and the indicator or decision variable with δ.

It is a frequently occuring situation when we want to distinguish between the state where $x = 0$ and the state when $x > 0$. For this we can use an indicator variable δ. By introducing the following constraint we can force δ to take value 1 when $x > 0$:

$$x - M\delta \leq 0, \tag{2.1}$$

where M is a constant coefficient representing a known upper bound for x.
With inequality (2.1) we now have logically the following condition:

$$x > 0 \rightarrow \delta = 1. \tag{2.2}$$

Now we would like to impose the other condition, that is

$$x = 0 \rightarrow \delta = 0, \tag{2.3}$$

or equivalently,

$$\delta = 1 \rightarrow x > 0. \tag{2.4}$$

Together condition (2.2) and condition (2.3) or (2.4) impose the following logical relation:

$$\delta = 1 \leftrightarrow x > 0. \tag{2.5}$$

It is not possible to represent (2.4) by a constraint. Instead we can find a realistic lower bound m for the continuous variable x. With this lower bound we can satisfy the condition:

$$\delta = 1 \rightarrow x \geq m. \tag{2.6}$$

For this we can use the following constraint:

$$x - m\delta \geq 0. \tag{2.7}$$

In the above described method we only use the decision variable δ to distinguish between two states of one continuous variable. It is possible to use indicator variables in a similar way to show whether an inequality holds or does not hold.

First, we wish to indicate whether the following inequality holds by means of an indicator variable δ:

$$\sum_j a_j x_j \leq b.$$

Therefore we would like to model the following condition first:

$$\delta = 1 \to \sum_j a_j x_j \leq b. \tag{2.8}$$

Condition (2.8) can be represented by a constraint:

$$\sum_j a_j x_j + M\delta \leq M + b. \tag{2.9}$$

The usual way of constructing (2.9) from condition (2.8) is the following method.

First, we would like to establish the logical condition, $\delta = 1 \to \sum_j a_j x_j \leq b$,
that is equivalent with the condition:
$\delta = 1 \to \sum_j a_j x_j - b \leq 0$.
In the other way it means, that if $(1 - \delta) = 0$ then we wish to have $\sum_j a_j x_j - b \leq 0$.
This condition is imposed if

$$\sum_j a_j x_j - b \leq M(1 - \delta),$$

where M is a sufficiently large upper bound, such that it does not give an undesired constraint.
Now the next step is to model the reverse implication of constraint (2.8), i.e.,

$$\sum_j a_j x_j \leq b \to \delta = 1. \tag{2.10}$$

This can be expressed with the following condition, which is equivalent with (2.10):

$$\delta = 0 \to \sum_j a_j x_j \nleq b, \tag{2.11}$$

i.e.,

$$\delta = 0 \to \sum_j a_j x_j > b. \tag{2.12}$$

When we are dealing with the expression $\sum_j a_j x_j > b$, we run into the same difficulties that we met with the expression $x > 0$ in condition (2.4).
We must rewrite the expression

$$\sum_j a_j x_j > b,$$

as

$$\sum_j a_j x_j \geq b + \epsilon,$$

where ϵ is some small tolerance value beyond which we will regard the constraint as having been broken.
If the coefficients a_j are integers as well as the variables x_j for all j, there is no difficulty as ϵ can be taken equal to 1.
With this ϵ (2.12) may now be written as:

$$\delta = 0 \rightarrow \sum_j a_j x_j - (b + \epsilon) \geq 0. \tag{2.13}$$

Now we can represent condition (2.13) with a constraint, which is:

$$\sum_j a_j x_j - (m - \epsilon)\delta \geq b + \epsilon, \tag{2.14}$$

where m is a lower bound for expression $\sum_j a_j x_j - b$.

Using the above described method for inequalities, which are of the form $\sum_j a_j x_j \leq b$, we can indicate whether an inequality of the form

$$\sum_j a_j x_j \geq b$$

holds or not.
We only have to transform the relation '$\geq$' into the relation '$\leq$'.
The logical condition

$$\delta = 1 \rightarrow \sum_j a_j x_j \geq b \tag{2.15}$$

can be represented with the inequality

$$\sum_j a_j x_j + m\delta \geq m + b, \tag{2.16}$$

where m is again the lower bound of the expression

$$\sum_j a_j x_j - b.$$

For the condition

$$\delta = 0 \rightarrow \sum_j a_j x_j < b \tag{2.17}$$

when applying the described method, we get the following constraint,

$$\sum_j a_j x_j - (M + \epsilon)\delta \leq b - \epsilon, \tag{2.18}$$

where M is again the upper bound for the expression $\sum_j a_j x_j - b$.

Finally, to use an indicator variable δ for a constraint of the form '=' such as

$$\sum_j a_j x_j = b,$$

is more complicated.
We can use $\delta = 1$ to indicate that the '$\leq$' and '$\geq$' cases hold simultaneously. This is done by stating both (2.9) and (2.16) together.
If $\delta = 0$ we want to force either the '$\leq$' or the '$\geq$' constraint to be broken.
This can be done by expressing (2.14) and (2.18) with two variables δ_1 and δ_2 giving

$$\sum_j a_j x_j - (m - \epsilon)\delta_1 \geq b + \epsilon, \tag{2.19}$$

$$\sum_j a_j x_j - (M + \epsilon)\delta_2 \leq b - \epsilon. \tag{2.20}$$

The indicator variable δ forces the required condition by the extra constraint

$$\delta_1 + \delta_2 - \delta \leq 1. \tag{2.21}$$

2.2 Logical constraints

In the previous section it was pointed out that extra 0-1 variables can be introduced into an LP or IP model as decision variables or indicator variables. In these models it has frequently happened that we have to model also the logical connections between these decision or indicator variables with linear programming tools. Including these connections in a mathematical programming model we will able to represent real life problems in a more efficient way.
In this section we will introduce some basic concepts and properties from mathematical logic, which are necessary from the modeling point of view.

It will be convenient to use notations from Boolean algebra. First of all we would like to introduce the so called set of connectives. In general, connectives are often used to make compound propositions. The main ones are shown in the following table (p and q are given propositions):

Name	**Symbol**	**Representation**	**Meaning**
negation	$\neg$	$\neg p$	'not p'
conjuction	$\wedge$	$p \wedge q$	' p and q'
disjunction	$\vee$	$p \vee q$	'p or q (or both)'
implication	$\rightarrow$	$p \rightarrow q$	'if p then q'
biconditional	$\leftrightarrow$	$p \leftrightarrow q$	'p if and only if q'

Table 2.1: Main compound propositions

The truth value of these compound propositions only depends on the truth value of its components. It is quite important to understand how the propositions value will change if the value of a component changes. These relations are shown in Table 2.2, where p, q, are given propositions, t is written if a statement is true and f if it is false.

p	q	$\neg p$	$p \wedge q$	$p \vee q$	$p \rightarrow q$	$p \leftrightarrow q$
t	t	f	t	t	t	t
t	f	f	f	t	f	f
f	t	t	f	t	t	f
f	f	t	f	f	t	t

Table 2.2: Truth table of main compound propositions

There are some equivalences ($\equiv$) between these propositions, that are sufficient for dealing with connectives. In the following, we will explore these equivalences.
Let p, q and r be given propositions, t stands for true and f stands for false statements. With these terms, we have the following laws:

Idempotent laws

$$p \vee p \equiv p \ , \quad p \wedge p \equiv p \, . \tag{2.22}$$

Commutative laws

$$p \vee q \equiv q \vee p \ , \quad p \wedge q \equiv q \wedge p \, . \tag{2.23}$$

Associative laws

$$(p \vee q) \vee r \equiv p \vee (q \vee r) \ , \quad (p \wedge q) \wedge r \equiv p \wedge (q \wedge r) \, . \tag{2.24}$$

Distributive laws

$$p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)\,, \tag{2.25}$$

$$p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)\,. \tag{2.26}$$

Domination laws

$$p \vee t \equiv t\;,\quad p \wedge t \equiv p\;,\quad p \vee f \equiv p\;,\quad p \wedge f \equiv f\;, \tag{2.27}$$

$$p \vee \neg p \equiv t,\quad p \wedge \neg p \equiv f\;,\quad \neg t \equiv f\;,\quad \neg f \equiv t\,. \tag{2.28}$$

Double negation law

$$\neg\neg p \equiv p\,. \tag{2.29}$$

De Morgan's laws

$$\neg(p \vee q) \equiv \neg p \wedge \neg q\;,\quad \neg(p \wedge q) \equiv \neg p \vee \neg q\,. \tag{2.30}$$

Implication laws

$$p \rightarrow q \equiv \neg p \vee q\,. \tag{2.31}$$

$$\neg(p \rightarrow q) \equiv \neg(\neg p \vee q) \equiv p \wedge \neg q\,, \tag{2.32}$$

$$\neg q \rightarrow \neg p \equiv \neg\neg q \vee \neg p \equiv p \rightarrow q\,, \tag{2.33}$$

$$p \rightarrow (q \wedge r) \equiv (p \rightarrow q) \wedge (p \rightarrow r)\,, \tag{2.34}$$

$$p \rightarrow (q \vee r) \equiv (p \rightarrow q) \vee (p \rightarrow r)\,, \tag{2.35}$$

$$(p \wedge q) \rightarrow r \equiv (p \rightarrow r) \vee (q \rightarrow r)\,, \tag{2.36}$$

$$(p \vee q) \rightarrow r \equiv (p \rightarrow r) \wedge (q \rightarrow r)\,. \tag{2.37}$$

2.3 The IP formulation of the logical relationships

In the previous sections we introduced indicator, decision variables to distinguish between certain states of the variables in an LP or IP model. Afterwards we dealt with some basic concepts from mathematical logic to give a background for manipulating with these variables.
In this section we would like to describe how we can model these Boolean algebraic manipulations with equalities and inequalities in an LP or IP modeling area.
From now on we will use X_i to stand for a proposition with truth value $\delta_i \in \{0,1\}$ i.e., X_i is true if and only if $\delta_i = 1$, where $\delta_i \in \{0,1\}$.
The following propositions and constraints are equivalent [9]:

$$X_1 \vee X_2 \quad \equiv \quad \delta_1 + \delta_2 \geq 1, \tag{2.38}$$

$$X_1 \wedge X_2 \quad \equiv \quad \delta_1\delta_2 = 1 \quad \equiv \quad \delta_1 = 1,\, \delta_2 = 1, \tag{2.39}$$

$$\neg X_1 \equiv \delta_1 = 0, \tag{2.40}$$

$$X_1 \rightarrow X_2 \equiv \delta_1 - \delta_2 \leq 0, \tag{2.41}$$

$$X_1 \leftrightarrow X_2 \equiv \delta_1 - \delta_2 = 0. \tag{2.42}$$

If a product term such as $\delta_1\delta_2$ were to appear anywhere in a model, the model could be made linear by the following steps:

(i) replace $\delta_1\delta_2$ by a $0-1$ variable δ_3,

(ii) impose the logical condition

$$\delta_3 = 1 \leftrightarrow \delta_1 = 1,\ \delta_2 = 1 \tag{2.43}$$

by means of the extra constraints

$$\begin{aligned} -\delta_1 + \delta_3 &\leq 0, \\ -\delta_2 + \delta_3 &\leq 0, \\ \delta_1 + \delta_2 - \delta_3 &\leq 1. \end{aligned} \tag{2.44}$$

It is even possible to linearize terms involving a product of a 0-1 variable δ with a continuous variable x. The term $x\delta$ can be treated in the following way:

(i) replace $x\delta$ by a continuous variable y ,

(ii) impose the logical condition

$$\begin{aligned} \delta = 0 &\rightarrow y = 0, \\ \delta = 1 &\rightarrow y = x. \end{aligned}$$

by the extra constraints

$$\begin{aligned} y - M\delta &\leq 0, \\ -x + y &\leq 0, \\ x - y + M\delta &\leq M, \end{aligned} \tag{2.45}$$

where M is an upper bound for x (and hence also y).

2.4 The model with logical constraints

So far in this chapter we established the mathematical background for representing logical relationships between variables in mathematical LP or IP programming. With this we will create a model in which these relations are involved.
In this section we will focus on the workforce scheduling problems. In our model we will use logical constraints to represent different restrictions (e.g., management policy, contractual obligations, governmental regulations, etc.). To deal with the model it is sufficient to introduce some new terms and definitions (the definitions from Section 1.1 will be used also here).

2.4.1 Terms, variables and sets

In the constraint set of the model we will use the following variables and sets:

$\mathcal{I}$	the index set of employees,
$\mathcal{K}$	the index set of different types employees,
$\mathcal{T}_k$	the index set of employees from type k, where $k \in \mathcal{K}$,
$\mathcal{P}$	the index set of periods,
$n_p^{(k)}$	the number of employees needed in period p from type k,
x_{pi}	$\begin{cases} 1, & \text{if employee } i \text{ is hired for period } p, \\ 0, & \text{if employee } i \text{ is not hired for period } p, \end{cases}$
$y_{pe}^{(k)}$	the number of extra employees for period p from type k.
$Logic(X)$	all the necessary logical constraints on matrix X, of which the elements are denoted by x_{pi}.

With index set $\mathcal{K}$ we can represent different types of employees, e.g., if there are different skills for handling different types of jobs. This situation occurs frequently for example in call centers (we will discuss this topic in Section 3.3). We use the index set $\mathcal{T}_k$ to group employees according to their skills. Set $\mathcal{P}$ was already defined and used in the previous models from literature in Chapter 1.
Because we have different types of employees, who can handle different jobs, we need to distinguish between required levels of staffing for all types. We will use parameter $n_p^{(k)}$ in our model to make this distinction between the required number of workers in all types. The variable $x_{pi} \in \{0, 1\}$ is for scheduling employees into periods.

We already made a distinction between employees with different skills. It is quite rational to differentiate employees who have a long term contract with the company and employees who are applying for a part time job with exact requirements for their working time, i.e., they have periods in which they can work and periods in which it is impossible for them to work. For this reason we will introduce index set $\mathcal{I}_c$ for the employees with a long term contract and index set $\mathcal{I}_p$ for employees with a part time job.
Notice that,

$$\begin{aligned} \mathcal{I}_c \cap \mathcal{I}_p &= \emptyset \\ &\text{and} \\ \mathcal{I}_c \cup \mathcal{I}_p &= \mathcal{I}. \end{aligned}$$

It is possible that we do not have enough employees to satisfy the required level (e.g., we do not have enough employees in set $\mathcal{I}_c$ and there are only a few members in $\mathcal{I}_p$). The companies usually try to avoid this situation but in case of unpredictably high load they have to hire extra workers (e.g., those who have a day off). In our model we denote with $y_{pe}^{(k)}$ the number of extra employees for period p from type k. These employees are far the most expensive ones so we will try to avoid to use them.

In our model we will use matrix A to represent the exact possible working times for employees with a part time job (i.e., employees in the index set $\mathcal{I}_p$):

$$a_{pi} \quad \begin{cases} 1, & \text{if employee } i \text{ can work in period } p, \\ 0, & \text{if for employee } i \text{ period } p \text{ is not a possible working period.} \end{cases}$$

2.4.2 The constraint set of the model

With the terms and notations from the previous section the constraint set of our scheduling model is the following:

$$\left.\begin{array}{l} A - X_1 \geq 0 \\ \sum\limits_{i\, \in \mathcal{I}_k} x_{pi} + y_{pe}^{(k)} \geq n_p^{(k)} \quad \forall k,\, p \\ Logic(X) \end{array}\right\} \mathbb{P}_{LC}.$$

Because we give indexes to employees it is possible to partition X into to submatrices, i.e., $X = [X_1\ X_2]$. In this partition X_1 is a sub-matrix of X, of which the columns are representing employees with a part time job, i.e., $X_1 \in \{0,1\}^{|\mathcal{P}|\times|\mathcal{I}_p|}$ and sub-matrix $X_2 \in \{0,1\}^{|\mathcal{P}|\times|\mathcal{I}_c|}$ is representing employees with a long term contract.
In our model we use constraint

$$A - X_1 \geq 0$$

to schedule only those part time employees to a period, who are able to work in that one. It is very important that this constraint has only effect on employees with a part time job (employees in index set $\mathcal{I}_p$), because we formulate their exact requirements for working time with it.
With the second constraint, i.e.,

$$\sum_{i\, \in \mathcal{I}_k} x_{pi} + y_{pe}^{(k)} \geq n_p^{(k)} \quad \forall k,\, p$$

we ensure that sufficient number of employees are present for every period from every type. This constraint is always satisfiable because of $y_{pe}^{(k)}$ and thus our $\mathbb{P}_{LC}$ is always feasible.
In the $Logic(X)$ part it will be useful if we put rarely or never changing constraints (e.g., governmental restrictions, special characteristics of the firm, contractual obligations, etc.).

2.4.3 The objective function

In the previous section we defined and introduced the constraints for our model. As in every mathematical programming model, we should define an objective function, with which we can express the goals of the model. This part of the modeling method is quite complicated because we should find the proper balance between our goals.
In this section we will focus on our objectives and the way we can involve them into the model.

Afterwards we will use $\mathcal{S}_{\mathbb{P}_{LC}}$ to denote the solution set of $\mathbb{P}_{LC}$, i.e.,

$$\mathcal{S}_{\mathbb{P}_{LC}} = \left\{ (X, y) \in \{0,1\}^{|\mathcal{P}|\times|\mathcal{I}|} \times \mathbb{N}^{|\mathcal{P}|\cdot|\mathcal{K}|} \;\middle|\; \begin{array}{l} A - X_1 \geq 0 \\ \sum\limits_{i \in \mathcal{T}_k} x_{pi} + y_{pe}^{(k)} \geq n_p^{(k)} \quad \forall k,\, p \\ Logic(X) \end{array} \right\}. \tag{2.46}$$

We would like to introduce a binary relation between the X_2 partitions of X in $\mathcal{S}_{\mathbb{P}_{LC}}$ to represent the preferences of employees with a long term contract (e.g., employees can prefer a schedule for themselves against another one).
We will use the following notation for this partition set:

$$\mathcal{S}^c_{\mathbb{P}_{LC}} := \left\{ Y \middle| X = \begin{bmatrix} X_1 \, Y \end{bmatrix},\, X \in \mathcal{S}_{\mathbb{P}_{LC}} \right\}.$$

There are some basic and rational properties of a preference relation $\preceq$, that we would like satisfy. These are the following:

Definition 2.4.1 *If $\preceq$ is a binary relation in $\mathcal{S}^c_{\mathbb{P}_{LC}}$, then we can define two other binary relations with*

$$Z \prec V \iff Z \preceq V \text{ and } V \npreceq Z$$

and

$$Z \sim V \iff Z \preceq V \text{ and } V \preceq Z,$$
$$\forall\ Z, V \in \mathcal{S}^c_{\mathbb{P}_{LC}}.$$

A binary relation $\preceq$ is transitive, if

$$Z \preceq V \text{ and } V \preceq W \implies Z \preceq W,$$
$$\forall\ Z, V, W \in \mathcal{S}^c_{\mathbb{P}_{LC}}.$$

A binary relation $\preceq$ is complete, if

$$Z \preceq V \text{ or } V \preceq Z,$$
$$\forall\ Z, V \in \mathcal{S}^c_{\mathbb{P}_{LC}}.$$

These properties are very natural if we want to model the preference of employees. With a preference relation we can distinguish between 'good' and 'bad' objects (schedules). This distinction is based on the subjective order of the decision maker. We can represent this order with a metric, which expresses the distance between an optimal schedule and any other schedules in $\mathcal{S}^c_{\mathbb{P}_{LC}}$.

We will denote with S^* the optimal schedule for employees with a long term contracts (i.e., all employees having their ideal schedule for the day). In this $S^* \in \{0,1\}^{|\mathcal{P}|\times|\mathcal{I}_c|}$ the columns represent the employees, the rows represent the periods (as in X). So

$$S^*_{pi} \quad \begin{cases} 1, & \text{if employee } i \text{ would like to work in period } p, \\ 0, & \text{if employee } i \text{ would not like to work in period } p. \end{cases}$$

With this ideal schedule we can measure the quality of any other schedules. The reasonable way to do this is to consider a schedule 'good' if it is 'near' to the ideal one and consider a schedule 'bad' if it is 'far' from the ideal one. It is important to see that S^* is not necessary an element of the set $\mathcal{S}^c_{\mathbb{P}_{LC}}$.
To deal with such notations as 'near' and 'far' we will introduce a metric between matrices in $\{0,1\}^{|\mathcal{P}|\times|\mathcal{I}_c|}$ space.

Definition 2.4.2 *A metric space $\mathcal{M} = (\mathcal{X}, d)$ consists of a set $\mathcal{X}$ together with a function, called metric, $d : \mathcal{X} \times \mathcal{X} \longrightarrow \mathbb{R}$ such that*

i) $d(x,y) \geq 0$ for all $x, y \in \mathcal{X}$, and $d(x,y) = 0 \Longleftrightarrow x = y$,

ii) $d(x,y) = d(y,x)$ for all $x, y \in \mathcal{X}$,

iii) $d(x,z) \leq d(x,y) + d(y,z)$ for all $x, y, z \in \mathcal{X}$.

As we can see, the definition of metric space allows us to define different kinds of metrics. For our purposes, we will use the so called discrete metric. This is defined by

$$d_0(x,y) = \begin{cases} 0, & \text{if } x = y, \\ 1, & \text{if } x \neq y. \end{cases}$$

Using this metric, we can define a metric in our $\{0,1\}^{|\mathcal{P}|\times|\mathcal{I}_c|}$ space, such that

$$d_M(X,Y) = \sum_{p,i} v_i d_0(X_{pi}, Y_{pi}),$$

where

v_i positive integer weight for column i
(in our model we will represent the value or the seniority of the employee with it).

Lemma 2.4.1 *The space* $\mathcal{M} = (\{0,1\}^{|\mathcal{P}|\times|\mathcal{I}_c|}, d_M)$ *is a metric space, with* $d_M : \{0,1\}^{|\mathcal{P}|\times|\mathcal{I}_c|} \times \{0,1\}^{|\mathcal{P}|\times|\mathcal{I}_c|} \longrightarrow \mathbb{N}$.

Proof of Lemma 2.4.1
To proof that d_M is a metric we should show that it satisfies all the properties in Definition 2.4.2.
First, we would like to prove property i) from the definition:

$$d_M(X,Y) = \sum_{p,i} v_i d_0(X_{pi}, Y_{pi})$$

and

$$v_i d_0(X_{pi}, Y_{pi}) \geq 0 \quad \forall\, p, i$$

thus, $d_M(X,Y) \geq 0$. The function $d_M(X,Y) = 0$ if and only if $v_i d_0(X_{pi}, Y_{pi}) = 0$ for all p, i, thus

$$d_M(X,Y) = 0 \text{ if and only if } X = Y,$$

which means that $d_M(X,Y)$ satisfies property i).

Now we would like to prove property ii), i.e., the symmetry:

$$d_M(X,Y) = \sum_{p,i} v_i d_0(X_{pi}, Y_{pi}) = \sum_{p,i} v_i d_0(Y_{pi}, X_{pi}) = d_M(Y,X),$$

which means that $d_M(X,Y)$ satisfies property ii).

Finally, we show that d_M satisfies property iii), i.e., it satisfies the triangle inequality:

$$d_M(X,Z) = \sum_{p,i} v_i d_0(X_{pi}, Z_{pi}) \leq \sum_{p,i} v_i \Big(d_0(X_{pi}, Y_{pi}) + d_0(Y_{pi}, Z_{pi}) \Big) =$$

$$= \sum_{p,i} v_i d_0(X_{pi}, Y_{pi}) + \sum_{p,i} v_i d_0(Y_{pi}, Z_{pi}) = d_M(X,Y) + d_M(Y,Z),$$

which means that $d_M(X,Y)$ satisfies property iii).
As we can see our d_M function satisfied all the sufficient properties, thus $d_M(X,Y)$ is a metric. It is also clear that $d_M(X,Y)$ satisfy the natural conditions in Definition 2.4.1.

So far we introduced a metric, with which we can represent the preferences of employees. Using this metric, one of our goal is to minimize the distance between the optimal schedule S^* and the solution, which means that we would like to give the 'best' possible schedule.

We have another very natural goal in our model. It is to minimize the cost of the schedule, i.e., minimize the cost of employees. For this we would like to introduce the following notations:

$c_{pe}^{(k)}$ the cost of extra employees for period p from type k,
c_{pi} the cost of employee i for period p, $i \in \mathcal{I}$.

With these notations our objective function is the following:

$$\min w_1 \left[\sum_{\substack{p \in \mathcal{P} \\ k \in \mathcal{K}}} c_{pe}^{(k)} y_{pe}^{(k)} + \sum_{\substack{i \in \mathcal{I} \\ p \in \mathcal{P}}} c_{pi} x_{pi} \right] + w_2 d_M(S^*, X_2). \tag{2.47}$$

In this objective function w_1 and w_2 are weights to determine the balance between the cost part and the preference part.

Finally, we have all elements to introduce our labor scheduling model, which is

$$\min w_1 \left[\sum_{\substack{p \in \mathcal{P} \\ k \in \mathcal{K}}} c_{pe}^{(k)} y_{pe}^{(k)} + \sum_{\substack{i \in \mathcal{I} \\ p \in \mathcal{P}}} c_{pi} x_{pi} \right] + w_2 d_M(S^*, X_2), \tag{2.48}$$

subject to

$$\left. \begin{array}{l} A - X_1 \geq 0 \\ \sum\limits_{i \in \mathcal{T}_k} x_{pi} + y_{pe}^{(k)} \geq n_p^{(k)} \quad \forall k,\, p \\ Logic(X) \end{array} \right\} \mathbb{P}_{LC}. \tag{2.49}$$

Chapter 3

Call centers

Up to this point we dealt with the basic staff scheduling models and we set up a model with logical constraints.
In this chapter we study the world of call centers; a real life area, where workforce scheduling plays a very important role.
First we will give a basic introduction about these centers.
After we will discuss some basic concepts from queueing theory; a field of mathematics, which helps us to model call centers.
Finally, we will go into details of workforce scheduling in call centers, to introduce this field of problems for the next chapter where we will solve a specific workforce scheduling problem.

3.1 Introduction

Nowadays, the most preferred and frequently used way for companies to communicate with their customers are call centers and their contemporary successors, contact centers. The economic role of these centers is thus significant and rapidly expanding.
Most organizations with customer contact - private companies, as well as government and emergency services - have re-engineered their infrastructure to communicate with customers via call centers.
For many companies operating world-wide, such as airlines, banks, credit card companies call centers provide a primary link between customer and service provider.

3.2 Call centers in general

From Koole [7]: "A call center can be considered as a set of resources - typically personnel, telecommunication equipment and computers - which enable to deliver different kind of services via the telephone".
More generally, a current trend is the extension of the call center into a contact center, which is a call center in which the traditional telephone service is enhanced by some

additional multi-media customer-contact channels, commonly e-mail, fax, Internet or chat.

3.3 Call center characteristics

Call centers can be categorized along many dimensions. Call centers with different characteristics can have huge differences from the mathematical point of view. Thus it is very important to determine the basic characteristics of our center before we start to build a mathematical model of it. The functions that call centers can provide vary highly: help desk for computer software, emergency response (e.g., police, ambulance etc.), costumer service, tele-marketing and order taking.
They greatly vary in size and geographic dispersion, from single-located centers with few agents that take local calls to large multi-site international centers in which thousands of agents work at any time.

Modern call centers are challenged with multitude types of calls, coming in over different communication channels (telephone, Internet, fax, e-mail, chat mobile devices, ...), thus these centers require agents with the skill to handle one or more types of calls (single- vs. multi-skilled agents). When multiple skills are required to handle calls, a center may cross-train low-skilled employees to handle all type of calls.
Due to the growing complexity of contact centers the organization of work becomes more complex. A recent trend in networking is skill based routing that is used to route calls to appropriate agents.

The central characteristic of a call center is whether it handles inbound or outbound calls. Inbound call centers handle incoming calls that are initiated by outside callers calling to a center. Typically, these types of centers provide costumer support, help desk services, reservation and sales support for airlines and hotels, and order-taking functions for catalog and web-based merchants.
Outbound call centers handle outgoing calls, calls that are initiated from within a center. These types of operations have been traditionally associated with advertisement, tele-marketing and survey businesses. A recent development in some inbound centers is to initiate outbound calls too.

3.4 Mathematics and call centers

In every call center the typical goal is to provide service at a given quality, subject to a given budget. Call center managers are thus faced with the problem of having a high service level with rapidly changing inbound calls. This service level is usually quantified in terms of congestion or other performance measures.

Because of this highly variable environment, the performance in terms of the qual-

ity of the service is always changing. For providing an acceptable service level during operation hours we have two possibilities.
The first possibility is to monitoring the performance and if it reaches an unacceptable level; for example, too many customers waiting or too many agents are idle, then react.
The second possibility is to build up a mathematical model for supporting the design or the control of the center.
In the first case the reaction is based on subjective experiences and in this case this could lead to a bad decision if the resulting performance is worse than expected. Using mathematical models, which are usually based on analytical approaches from Operations Research and Queueing Theory, we can control the performance between acceptable levels. These models have become integral parts of the widely used workforce scheduling tools.

3.5 Queueing models in call centers

In the previous section we discussed the importance of using mathematical models in call centers. In this section we deal with these queueing models.

3.5.1 The modeling area

In the service area, where customers and service providers establish contact, we can often face queues and queueing problems. Call centers thus can be viewed, naturally and usefully as queueing systems. This is clear from Figure 3.1 which is a simple call center operational scheme.
In this queueing system we have the following elements:

arrivals - group of customers those whose are calling the center

queue - a queue of customers those whose are waiting for the service

agents - call center employees who respond to calls

busy signals - if all lines are busy and there is no space in the queue of callers

abandonment - if the customer does not want to wait in the queue anymore

retrials - if a customer tries to contact the center immediately after he/she had a busy line or an abandonment

lost calls - if a customer leaves the call center without being served and he/she does not contact immediately after it.

Although this is an operational scheme of a call center, modern call (contact) centers have more complex characteristics as we already discussed it in Section 3.3. These centers are thus more complicated queueing networks and modeling them is a difficult problem full of challenges.

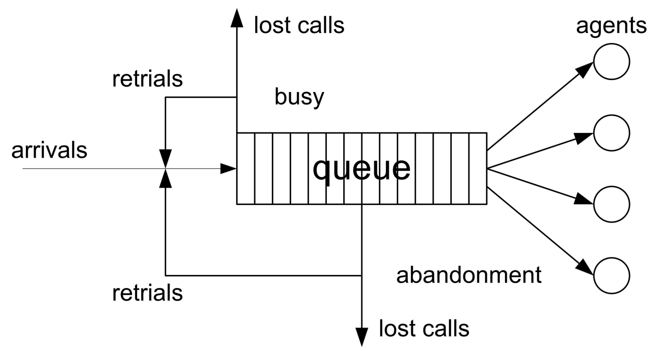

Figure 3.1: Call center operational scheme

3.5.2 Queueing models

In this subsection we would like to lay down the basic mathematical background of queueing theory. The basic queueing model is shown in Figure 3.2.

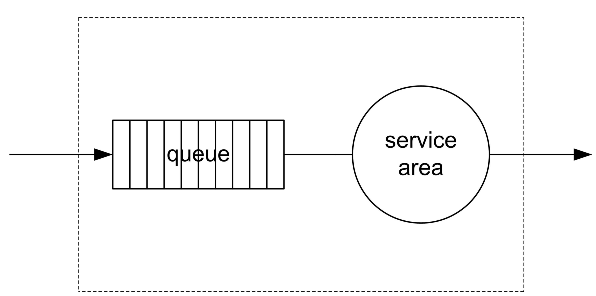

Figure 3.2: Basic queueing system

There is a wide variety of different queueing models, but there are some basic characteristics with which we can categorize different queueing models. These characteristics are the following:

- *Arrival process of the customers*
  The customers arrive to the queueing system in random fashion and in different style.

  One by one or in batches : Customers can arrive one by one to the system or in batches.

Markovian arrivals : Customers arrive to the system according to Poisson process.

General arrivals : Customers arrive to the system according to general distribution, i.e., interarrival times have a general i.i.d distribution

- *Attitude of the customers*
  The customer behavior is varying in the queue.

  Infinite patience : Customers will wait in the queue until they will be served.

  Abandonment : Customers will leave after a while from the queue.

  Retrials : Customers, who did not receive service, will return to the queue just after they left it.

- *Service time*
  The service time of any customer in the system is also random.

  Markovian : The service time of any customer in the system is according to an exponential distribution.

  General : The service time of any customer in the system is according to a general distribution.

- *Service discipline*
  There are different ways of how the customers can be served in different models.

  FCFS : (first-come-first-served) customers are served in the order of arrival.

  LIFO: (last-in-first-out) the last customer who arrived is served first.

  PS: (processor sharing) all servers are sharing their capacity equally among the costumers.

  Priority Disciplines: every customer has a (static or dynamic) priority, the customer with the highest priority will be selected. This scheme can use preemption or not.

  Random: the service is in random order.

  Round Robin: every customer gets a time slice. If his/her service is not completed, he/she will re-enter the queue.

- *Service capacity*
  The number of servers can be different in different models.

  - *Single server model.*
  - *Multiple server model.*

- *Waiting places*
  The number of waiting places can be different in different models.

- *Infinitely many waiting spaces.*
- *Finitely many waiting spaces.*

Hereafter, we will use Kendall's notation for queueing models, which is of the form:

$$A|B|c|d,$$

where

A represents the arrival process;
B represents the service time;
c represents the number of servers (agents in call centers);
d represents the total number of places in the queueing system.

A is usually either M(Markovian) for Poisson arrivals or G for general arrival distribution. We will denote the average number of customers entering the system per time unit with λ.
B is usually either M(Markovian) for exponential service times or G for general service times. We will denote with β the average service time.
It is not surprising that c an d are integers with $c \leq d$ (i.e., $d = c+$ waiting spaces) and if $d = \infty$ then it is not shown in the notation.

Afterwards we will use the following terminology for queue lengths and waiting times:

L_Q is the stationary number of customers in the queue.
L is the stationary number of customers in the system.
W_Q is the time that an arbitrary customer spends waiting before service, in a stationary situation.
W is the time that an arbitrary customer spends in the system, while waiting and while being served.

3.5.3 Little's Law

Little's Law is one of the basic theorems in queueing theory. It gives a very important relation between the mean number of customers in the system, the mean response time and the average number of customers entering the system per time unit. Little's Law is the following:

$$E(L) = \lambda E(W).$$

In this case we applied Little's Law to the whole system. If we apply it only to the queue we get the following equation:

$$E(L_Q) = \lambda E(W_Q).$$

3.5.4 PASTA property

For queueing systems with Poisson arrivals ($M/\cdot/\cdot/\cdot$) a special property holds, called PASTA (Poisson arrivals see the time average). This property says that arriving customers find on average the system in the same situation as an outside observer at every arbitrary time point, i.e., the fraction of customers finding the system in a state α is the same as the fraction of time that the queueing system is in α. This property is quite important if we deal with these kind of queues.

3.5.5 The $M/M/s$ queue

For the applications, one of the most important queueing model is the $M/M/s$ queue. Because of this, we will make a deeper analysis of it.
In this model the arrival process is Poisson with rate λ. The service time is exponential with rate $\mu = \frac{1}{\beta}$ and the number of servers (agents) is s, where β is the average service time. This system can be modeled as a birth-death process in $\{0, 1, \dots\}$, where state j represents the number of customers in the system (including the customers in service). Denote with $a = \frac{\lambda}{\mu}$ the offered load in Erlang, with $\rho = \frac{a}{s}$ the load in Erlang per servers and with q_{ij} the transition rate from state i to state j. In our birth-death process:

$$q_{ij} = \begin{cases} \lambda, & \text{if } j = i+1, \\ \min\{i, s\}\mu, & \text{if } j = i-1, \\ 0, & \text{otherwise.} \end{cases}$$

The flow diagram of this queueing system is shown in Figure 3.3.

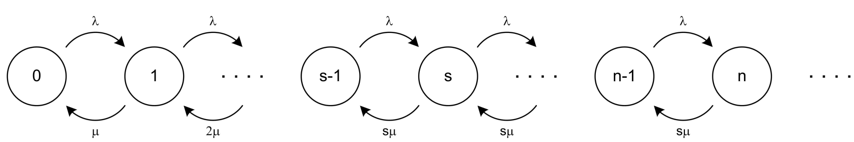

Figure 3.3: Flow digram for $M/M/s$ queue

Using PASTA property, the equilibrium equations are the following:

$$\begin{aligned} \lambda\pi_0 &= \mu\pi_1, \\ (\lambda + \min\{i, s\}\mu)\,\pi_i &= \lambda\pi_{i-1} + \min\{i+1, s\}\mu\pi_{i+1}, \qquad i > 0. \end{aligned}$$

In these equations π_i is the stationary probability, i.e.,

$$\pi_i = \lim_{t\to\infty} P(N(t) = i), \qquad i = 0, 1, \dots, \infty \ ,$$

where $N(t)$ is the number of costumers at time t in the system.

From the equilibrium equations we get that:

$$\pi_i = \begin{cases} \frac{a^i}{i!}\pi_0, & \text{if } i < s, \\ \frac{a^i}{s!s^{i-s}}\pi_0, & \text{otherwise.} \end{cases}$$

The value of π_0 can be derived from

$$\sum_{i=0}^{\infty} \pi_i = 1, \tag{3.1}$$

giving

$$\pi_0 = \left[\sum_{i=0}^{s-1} \frac{a^i}{i!} + \frac{a^s}{(s-1)!(s-a)}\right]^{-1}. \tag{3.2}$$

An important probability is the probability of delay:

$$C(s,a) := P(W_Q > 0) = \sum_{i=s}^{\infty} \pi_i = \tag{3.3}$$

$$= \pi_s \left[1 + \rho + \rho^2 + \dots\right] = \frac{\pi_s}{1-\rho} = \tag{3.4}$$

$$= \sum_{i=s}^{\infty} \frac{a^i}{s!s^{i-s}}\pi_0 = \sum_{j=0}^{\infty} \frac{a^{j+s}}{s!s^j}\pi_0 = \sum_{j=0}^{\infty} \frac{a^s}{s!}\frac{a^j}{s^j}\pi_0 = \tag{3.5}$$

$$= \frac{a^s}{(s-1)!(s-a)}\left[\sum_{i=0}^{s-1} \frac{a^i}{i!} + \frac{a^s}{(s-1)!(s-a)}\right]^{-1} = \tag{3.6}$$

$$= \frac{\frac{a^s}{s!(1-\rho)}}{\sum_{k=0}^{s-1} \frac{a^k}{k!} + \frac{a^s}{s!(1-\rho)}}. \tag{3.7}$$

Equation (3.7) is the famous Erlang delay (or Erlang C) formula.
It is very important to see that in equations (3.4) and (3.5) above we supposed that $\rho = \frac{a}{s} < 1$.
This is not only a mathematical fact, but a very natural assumption: if on average there are more arrivals than departures per time unit then the number of waiting calls increases to infinity.
With $C(s,a)$ we can get the following:

$$E(L_Q) = \sum_{i=0}^{\infty} i\pi_{s+i} = \frac{\pi_s}{1-\rho}\sum_{i=0}^{\infty}(1-\rho)i\rho^i = \tag{3.8}$$

$$= C(s,a)\sum_{i=0}^{\infty}(1-\rho)i\rho^i = C(s,a)\frac{\rho}{1-\rho}. \tag{3.9}$$

So we have that

$$E(L_Q) = \frac{\lambda C(s,a)}{s\mu - \lambda}. \tag{3.10}$$

From result (3.10) we can easily get $E(W_Q)$ by using Little's Law from Subsection (3.5.3);

$$E(W_Q) = \frac{C(s,a)}{s\mu - \lambda}. \tag{3.11}$$

Now using the equation, $E(W) = E(W_Q) + \frac{1}{\mu}$ we can get the following results:

$$E(W) = \frac{C(s,a)}{s\mu - \lambda} + \frac{1}{\mu}. \tag{3.12}$$

If we apply again Little's Law for the whole system, the expected queue length is

$$E(L) = \frac{\lambda C(s,a)}{s\mu - \lambda} + \frac{\lambda}{\mu}. \tag{3.13}$$

3.5.6 The $M/M/s/N$ queue

Another very important queueing model is the $M/M/s/N$ queue, where the arrival process and the service times are both Poisson with rate λ and $\mu = \frac{1}{\beta}$ respectively, where β is the average service time. In this model there are finitely many places in the system, denoted by N and the number of servers is s.
Denote again with $a = \frac{\lambda}{\mu}$ the offered load in Erlang, with $\rho = \frac{a}{s}$ the load in Erlang per servers and with q_{ij} the transition rate from state i to state j. With this notation the transition rate is the following:

$$q_{ij} = \begin{cases} \lambda, & \text{if } j = i+1, \quad i = 0,\dots,N-1\ , \\ \min\{i,s\}\mu, & \text{if } j = i-1, \quad i = 1,\dots,N\ , \\ 0, & \text{otherwise.} \end{cases}$$

Using these transition rates, the flow diagram of the $M/M/s/N$ queue is in Figure 3.4.
To calculate the stationary probabilities we use again the equilibrium equations. In this case these are the following:

$$\begin{aligned} \lambda\pi_0 &= \mu\pi_1, \\ (\lambda + \min\{i,s\}\mu)\,\pi_i &= \lambda\pi_{i-1} + \min\{i+1,s\}\mu\pi_{i+1} \qquad i = 1,\dots,N-1, \\ \lambda\pi_{N-1} &= s\mu\pi_N\ . \end{aligned}$$

From these equations we can get that:

$$\pi_i = \begin{cases} \dfrac{a^i}{i!}\pi_0, & \text{if } i < s, \\ \dfrac{a^i}{s!s^{i-s}}\pi_0, & i = s,\dots,N\ . \end{cases}$$

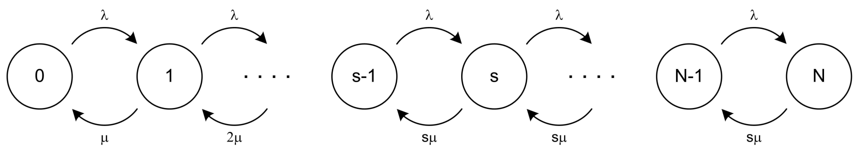

Figure 3.4: Flow digram of $M/M/s/N$ queue

Using the normalization equation, which is:

$$\sum_{i=0}^{N} \pi_i = 1, \tag{3.14}$$

we get that

$$\pi_0 = \left[\sum_{i=0}^{s-1} \frac{a^i}{i!} + \sum_{i=s}^{N} \frac{a^i}{s! s^{i-s}}\right]^{-1} = \tag{3.15}$$

$$= \left[\sum_{i=0}^{s-1} \frac{a^i}{i!} + \frac{a^s}{s!} \frac{\rho^{(N-s+1)} - 1}{\rho - 1}\right]^{-1}. \tag{3.16}$$

Now we can compute the probability of delay, which is

$$B(s,a) := P(W_Q > 0) = \sum_{i=s}^{N} \pi_i = \tag{3.17}$$

$$= \pi_s \left[1 + \rho + \rho^2 + \cdots + \rho^{N-s}\right] = \pi_s \frac{\rho^{(N-s+1)} - 1}{\rho - 1} = \tag{3.18}$$

$$= \frac{\rho^{(N-s+1)} - 1}{\rho - 1} \frac{a^s}{s!} \left[\sum_{i=0}^{s-1} \frac{a^i}{i!} + \frac{a^s}{s!} \frac{\rho^{(N-s+1)} - 1}{\rho - 1}\right]^{-1}. \tag{3.19}$$

With these probabilities we can compute the following equations:

$$E(L_Q) = \sum_{i=0}^{N-s} i\pi_{s+i} = \pi_s \frac{\rho^{(N-s+1)} - 1}{\rho - 1} \sum_{i=0}^{N-s} \frac{\rho - 1}{\rho^{(N-s+1)} - 1} i\rho^i = \tag{3.20}$$

$$= B(s,a) \sum_{i=0}^{N-s} \frac{\rho - 1}{\rho^{(N-s+1)} - 1} i\rho^i = \tag{3.21}$$

$$= B(s,a) \frac{\rho^{(N-s+1)}((N-s+1)\rho - N + s - 1 - \rho) + \rho}{\rho - 1}. \tag{3.22}$$

From this, we can compute $E(W_Q)$ with Little's Law,

$$E(W_Q) = B(s,a)\frac{\rho^{(N-s+1)}((N-s+1)\rho - N + s - 1 - \rho) + \rho}{\lambda(\rho-1)}. \tag{3.23}$$

Using again, that $E(W) = E(W_Q) + \frac{1}{\mu}$ we have that

$$E(W) = B(s,a)\frac{\rho^{(N-s+1)}((N-s+1)\rho - N + s - 1 - \rho) + \rho}{\lambda(\rho-1)} + \frac{1}{\mu}, \tag{3.24}$$

and

$$E(L) = B(s,a)\frac{\rho^{(N-s+1)}((N-s+1)\rho - N + s - 1 - \rho) + \rho}{\rho-1} + \frac{\lambda}{\mu}. \tag{3.25}$$

3.6 Workforce scheduling in call centers

Workforce scheduling is a problem, which frequently occurs in call centers.
The aim of it in these centers is to provide a proper service level, subject to a specified budget. The fact that salaries account 60-70% of the total operating cost of a call center shows that personnel scheduling is an important topic in these centers.

In the previous chapters we saw that such a center can be modeled as a queueing system. In case of call centers this model can be very complicated not only because of the complexity of a center but also due to the fluctuations of the calls (i.e., the load of the call center highly depends on time).
In Figure 3.5 we plotted the number of incoming calls per period in a call center. In this plot one period is 30 minutes long and the operating hours are from 8 am to 8 pm. The three different plots represent different days of a week: Monday with normal line, Wednesday with dotted line and Friday with dashed line.
As we can see the number of incoming calls not only differs from period to period but also from day to day.

For measuring the service level in this environment it is common to use one of the formulas from the previous chapter (equation (3.10)-(3.13) for $M/M/s$ queues and equations (3.20)-(3.25) for $M/M/s/N$ queues). From these equations it is clear that the number of agents has effect on these performance metrics.
In general we can say that all performance metrics are base on the Erlang formula or on one of its generalizations and thus these metrics are highly dependent on the agent number as well. To use one of these formulas we should estimate the load of the system. For this a common method is to divide the operating day into small periods (e.g., 15 min., 30 min. or 1 hour) and calculate the aggregate load for each period. This can be done by using historical data and mathematical tools. Based on these forecasted loads we can compute the minimum number of agents for each period to reach the minimum service level.

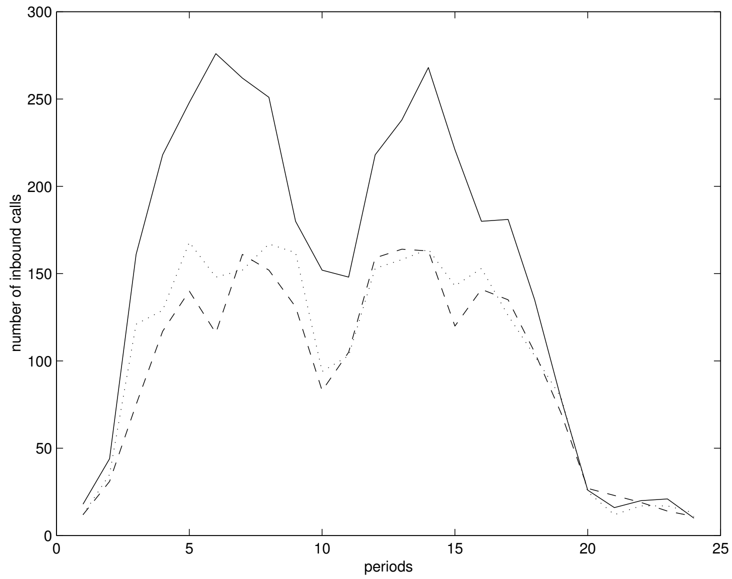

Figure 3.5: Number of inbound calls in a call center

The result of this is the input of the next step, i.e., to schedule employees into shifts providing the minimum number of agents in all operational periods.

In the next chapter we will solve a scheduling problem in a call center using the above described method with our scheduling model, which we have already described in Chapter 2.

Chapter 4

An application in call centers

Up to this point we described the basic background for solving an agent scheduling problem in call centers. In Chapter 1 we introduced the shift scheduling and the staff assigning problem and we analyzed some models from the literature. In Chapter 2 we introduced and built up our workforce scheduling model, in which we model implicitly shifts, i.e, with logical constraints. In Chapter 3 we gave a short survey about call centers and queueing theory.

In this chapter we will solve an agent scheduling problem in call centers using our model in Chapter 2 on page 29.

4.1 The scheduling environment

The scheduling environment chosen for this problem involves a continuous 8-hour work day from 9.00 to 17.00. Employee requirements are determined for 32 15-minute periods i.e., our set $\mathcal{P} = \{1, 2, 3, \ldots, 31, 32\}$. In our model we have three different types of employees: employees with a long term contract, part time employees and extra employees, where all of the employees have single skill, i.e., $\mathcal{K} = \{1\}$. We have different restrictions for each type of workers.
For employees with a long term contract we have the following restrictions:

- they have to work in 6 hours shifts continually (including breaks),
- each of them is given exactly one half-hour lunch break (2 consecutive periods) and one 15-minute coffee break,
- they have two separate break windows specified as follows:
  the start time of the lunch break window is specified as two hours after the starting period of the operating hours i.e.,
  the lunch break window is from 11.00 to 13.00,
- the coffee break window is from 14.00 to 16.00,
- if an employee starts his/her work in one of the break windows then he/she will not receive break in that break window.

For part time employees we have possibilities for working, which were set up by themselves, e.g., in case of call centers it is common that they fill in a form in the Internet and mark out exact periods, in which they can work. Moreover, they have to work continually in their working periods too.

For extra employees we do not have any constraints. We can interpret them in two different ways.
At the one hand, we can say that they are real employees with a rather high salary for a period and we only hire them for those periods, in which we could not satisfy the requirement. In other hand, we can interpret them as auxiliary variables; with them our model is always feasible. Because in our model the goal is to minimize the value of the objective function, we can avoid to use these employees by giving high cost to them (in our test problem we will follow this interpretation). In this way we can discover understaffing and avoid it.

4.2 Input parameters

In our model there are input parameters, which we already described in Chapter 2. These parameters for our scheduling problem, which are contained in
Table 4.1, are the following:

- the indexes of periods in the first column,
- λ_p, i.e.,the number of call arrivals to the call center per minute for each period in the second column,
- the number of required agents for each period in column three,
- the exact requirements of part time employees for their working time, i.e., matrix A in columns 4-6,
- the optimal schedule for each employees with a long term contract, i.e., matrix S^* in columns 7-19.

In our model we used consistent cost parameters for full time employees, part time employees and extra employees respectively. This means that

$$\begin{aligned} c_{pe} = &\ 100 \quad \forall\, p \in \mathcal{P}, \\ c_{pi} = &\ 2 \quad\ \ \forall\, p \in \mathcal{P},\ i \in \mathcal{I}_c, \\ c_{pi} = &\ 1 \quad\ \ \forall\, p \in \mathcal{P},\ i \in \mathcal{I}_p. \end{aligned}$$

Furthermore, we used the 80-20 service level metric, with 25 sec. average service time (i.e., $\beta = 25$ sec.) to calculate the number of required agents from the number of incoming calls. This metric is defined as the following: 80% of costumers should be served before 20 sec., i.e.,

$$P(\text{wait} < 20 \text{ sec.}) = P(W_Q < 20) = 1 - C(s,a)e^{-(s-a)\frac{20}{\beta}} = 0.8\,,$$

where all of the notations were described in Chapter 3. To calculate these numbers we used the Erlang C calculator (http://www.math.vu.nl/ koole/ccmath/).

			Part time workers			**Full time workers**												
Iop[1]	**Apm**[2]	**Nora**[3]	I.	II.	III.	1	2	3	4	5	6	7	8	9	10	11	12	13
1	1	2	1	0	1	1	0	1	0	0	0	0	0	0	0	1	1	0
2	2	2	1	0	1	1	0	1	0	0	0	0	0	0	0	1	1	0
3	2	2	1	1	1	1	0	1	0	0	0	0	0	0	0	1	1	0
4	4	3	1	1	1	1	0	1	0	0	0	0	0	0	0	1	1	0
5	15	8	1	1	1	1	0	1	1	1	0	0	0	0	0	1	1	0
6	22	11	1	1	1	1	0	1	1	1	0	0	0	1	0	1	1	0
7	24	12	1	1	1	1	0	1	1	1	0	0	0	1	0	1	1	0
8	27	13	1	1	1	1	0	1	1	1	1	0	0	1	0	1	1	1
9	22	11	1	1	1	1	0	1	1	1	1	0	0	1	0	1	1	1
10	19	10	1	1	0	1	0	1	1	1	1	0	0	1	0	1	1	1
11	25	12	1	1	1	1	0	1	1	1	1	0	0	1	0	1	1	1
12	28	14	1	1	1	1	1	1	1	1	1	0	0	1	0	1	1	1
13	24	12	1	1	1	1	1	1	1	1	1	0	0	1	0	1	1	1
14	20	10	1	1	1	1	1	1	1	1	1	0	1	1	0	1	1	1
15	15	8	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1
16	15	8	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
17	24	12	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
18	25	12	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
19	31	15	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
20	26	13	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
21	29	14	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
22	23	11	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
23	17	9	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1
24	24	12	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1
25	20	10	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1
26	17	9	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	0
27	18	9	0	0	0	0	1	1	1	0	1	1	1	1	1	1	1	0
28	9	5	0	0	0	0	1	1	1	0	1	1	1	1	1	0	0	0
29	10	6	0	0	0	0	1	1	0	0	1	1	1	0	1	0	0	0
30	7	4	0	0	0	0	1	1	0	0	0	1	1	0	1	0	0	0
31	6	4	0	0	0	0	1	1	0	0	0	1	1	0	1	0	0	0
32	1	2	0	0	0	0	1	0	0	0	0	1	1	0	1	0	0	0

[1] Index of periods
[2] Arrivals per minute
[3] Number of required agents

Table 4.1: Input parameters of the problem

4.3 The implementation of the model

In this section we deal with the way how we implemented our labor scheduling model. We describe techniques to represent nonlinear terms and to model different conditions with logical constraints in our integer programming model.

4.3.1 The objective function in the model

In Subsection 2.4.3 we described the objective function of our model. From the definition of $d_0(x, y)$ it is clear that the d_M metric, which we use to express the preferences of employees, is not linear. Even so, it is possible to represent this metric in an IP model. In the case when both x and y are $0, 1$ variables, the discrete metric is

$$d_0(x, y) = |x - y| = \begin{cases} 0, & \text{if } x = y, \\ 1, & \text{if } x \neq y. \end{cases}$$

Using this result we get that $d_M(X, Y) = \sum_{p,i} v_i d_0(X_{pi}, Y_{pi}) = \sum_{p,i} v_i |X_{pi} - Y_{pi}|$. One of the commonly accepted ways to deal with absolute value function $|x - y|$ in IP is the following:

$$\min \delta, \tag{4.1}$$

subject to

$$x - y \leq \delta, \tag{4.2}$$

$$y - x \leq \delta, \tag{4.3}$$

$$x,\ y,\ \delta\ \in \{0, 1\}\,. \tag{4.4}$$

We can see that in this representation δ is equivalent with the expression $|x - y|$. In our model we followed this way to deal with the absolute value function.

4.3.2 Logical constraints in the model

One of the basic constraints is that all employees has to work continuously. For part time employees we can implement this constraint with the following logical condition: if an employee does not work in a period and he/she worked in the previous one then he/she cannot work in the forthcoming periods. We can represent this constraints with expression (2.9).

For full time employees we also have this constraint but for them the implementation is different because they have to work in 6 hours long shifts. Because of this, if they work in one period before the first break window then they have to work continuously until the beginning of the break window. We can model this condition with (2.16). We have to model breaks for these employees, i.e., the first and the second relief breaks. We can do it with establish the following logical condition: if an employee worked before the first or the second break window than he/she has to have break in the break window.

We can represent this constraint with using together inequality (2.9) and (2.16). For the lunch break we had to model also that its duration is 2 consecutive periods.
Taken all round these are the main logical representations, which we used to implement our labor scheduling model.

4.4 Experimental results with the test problem

In this section we deal with the experimental results of our scheduling problem.
We run different test problems with different combinations of weights, i.e., for w_1, w_2 and for v_i. All of these problems were formulated in GAMS and we solved them using the mixed integer programming option of GAMS with the OSL solver on a Sun Sparcstation with an Unix Solaris operating system. We created a formatted output from the GAMS output to make the analysis easier. In this formated output we display matrix $X = [X_1\ X_2]$, the number of extra employees per periods, the number of agents in periods and the number of required agents in periods. The value of the objective function is also shown in our outputs. We divided this value into two parts; one is the cost of the schedule and one is the distance of the schedule from the ideal schedule S^*. Both values are without the weight parameters w_1 and w_2.

In the first test problem the weights were the following:

$$
\begin{aligned}
w_1 &= 1, \\
w_2 &= 0, \\
v_i &= 1,\ i = 1, \ldots, 13.
\end{aligned}
$$

The result of this problem is shown in Table 4.3.
With these parameters we wanted to minimize the cost of the schedule without paying any attention to the satisfaction of the agents, i.e., we ignored the metric part of the objective function by choosing $w_2 = 0$. It is clear from the output that we do not need an extra employee in any period, which means that we satisfied the staffing requirements.
The maximum cost of any schedule is $100 \sum_p n_p = 28500$. This is the case when we only hire extra employees for all periods. The cost of our schedule is 593, which is approximately 2% of the worst case. It is important to see that our metric depends also on the values of v_i, thus it is possible that for different v_i we have distance in a different order of magnitude. The maximum distance between the ideal schedule S^* and the schedule with these parameters is $13 \times 32 = 416$ and our result is 120, which is approximately 29%. This ratio is not surprising because we ignored this part with the weight parameters.

In the second test problem we wanted to involve the preferences of the full time employees, thus we chose the weights as

$$\begin{aligned} w_1 &= 1, \\ w_2 &= 1, \\ v_i &= 1,\ i = 1, \ldots, 13. \end{aligned}$$

The result of this test problem is shown in Table 4.4. In this result we also do not have under-staffing and we can see the big effect of the weights. We involved the metric part in the model with the same weight as the cost part and our result, in sense of agents satisfaction, increase dramatically while the cost increased only slightly. The distance between the ideal schedule and our result is 88 and the cost of the schedule is 595 in this case, which means that we improve the satisfaction by 8%, i.e., the new result is 21% and the cost is only 595. In this test problem we can see the big effect of the weight parameters, with which we improved our results.

In the third test problem we wanted to make a distinction between full time employees. For this reason we changed the weight parameters of the problem. The parameters in this case are

$$\begin{aligned} w_1 &= 1, \\ w_2 &= 1, \\ v_i &= 1,\ \forall\, i \neq 1, 2, 3\,, \\ v_1 &= 2, \\ v_2 &= 2, \\ v_3 &= 2. \end{aligned}$$

The result of the problem is in Table 4.5. In this case we avoid also to hire extra employees and the cost of the schedule is 591. Because we changed the values of v_1, v_2 and v_3 our metric result is not really comparable with the previous ones. Instead of comparing the distance we can compare the number of no-hits in the results. This means that we can calculate the number of non-identical elements between the solution and the ideal schedule. This numbers are 120, 88 and 78 for the first, second and third test problem respectively, which means that we improved the satisfaction of our employees by changing the parameters. It is interesting to see that we not only improved the satisfaction of the employees, but with these parameters we were able to reduce the cost by 4, i.e., from 595 to 591.

In our fourth test problem we put larger accent to the metric part of the objective function, which means that we wanted to satisfy the employees more than before. The parameters of this problem are the following:

$$\begin{aligned} w_1 &= 1, \\ w_2 &= 5, \\ v_i &= 1,\ i = 1, \dots, 13. \end{aligned}$$

The result of this test problem is shown in Table 4.6. In this case we made a schedule without under-staffing, with cost 593. It is clear from the output that we succeeded in improving the satisfaction of the employees, i.e., we got 74 non-identical element in the schedule, which is approximately 18%. In this case we can have nearly the same conclusion as in the previous problem, namely that with these parameters we reduced the number of non-identical elements between the ideal schedule and our result, meanwhile the cost of the schedule remained approximately the same.

In our last test problem we used the following weight parameters:

$$\begin{aligned} w_1 &= 1, \\ w_2 &= 5, \\ v_i &= 1,\ \forall\, i \neq 1, 2, 3\,, \\ v_1 &= 2, \\ v_2 &= 2, \\ v_3 &= 2. \end{aligned}$$

The result of this problem is in Table 4.7. The cost of this schedule is 595, the distance from the ideal schedule is 97 and we avoid to hire extra employees again. It is also hard to compare the metric result with the other results because of the different v_i weights. Even so we can compare the distance with the result in the third test problem because all v_i are the same. In that case the distance was 101, which means that we improved again this performance measure.
So far we made an analysis of the test problems and discussed some results. In Table 4.2 all of the results are shown to make the comparison easier.

Problem	w_1	w_2	v_i	Cost of schedule	Number of non-identical elements			
					all	Employee 1	Employee 2	Employee 3
1	1	0	id.[1]	593	120	19	14	10
2	1	1	id.[1]	595	88	15	10	10
3	1	1	non-id.[2]	591	78	11	2	10
4	1	5	id.[1]	593	74	11	4	10
5	1	5	non-id.[2]	595	74	11	2	10

[1] $v_i = 1,\ i = 1, \dots, 13.$
[2] $v_i = 1,\ \forall\, i \neq 1, 2, 3\,,\ v_1 = 2\,,\ v_2 = 2,\ v_3 = 2$

Table 4.2: Results of the test problems

As we can see from Table 4.2 the cost of the schedules remained approximately the same during our experiments but meanwhile we made an improvement in the satisfactory level. These results show us again how important it is to find a good combination of weights. In the third test problem, where we made a distinction between employees meanwhile w_1, w_2 remained the same, we improved the satisfaction of the employees by 10 and we improved the first and the second employee's satisfaction with 4 and 8 respectively, which means that we could find a right balance using these weights.
In the last two cases, i.e., when we chose $w_2 = 5$, the number of non-identical elements between the ideal schedule and the result was 74 in both cases, which means that we increase significantly the satisfaction of our employees.

After this analysis, in the next chapter we will make some concluding remarks about our modeling method, computational results and we will mention some future prospects.

```
                     Workforce scheduling model

rows - index of periods
1-13 - index of full time employees
I.,II.,III. - index of part time employees
EXTRA - number of extra employees in periods
AGENTS - number of agents in periods
NORA - number of required agents in periods

Cost of Schedule   Distance from the optimal schedule
593                120
```

	1	2	3	4	5	6	7	8	9	10	11	12	13	I.	II.	III.	EXTRA	AGENTS	NORA
P1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	2	2
P2	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	2	2
P3	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	2	2
P4	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	3	3
P5	0	1	0	1	1	0	1	0	0	1	1	1	0	1	1	1	0	10	8
P6	0	1	1	1	1	1	1	0	0	1	1	1	0	1	1	1	0	12	11
P7	0	1	1	1	1	1	1	0	0	1	1	1	0	1	1	1	0	12	12
P8	0	1	1	1	1	1	1	0	1	1	1	1	0	1	1	1	0	13	13
P9	0	1	1	1	0	1	1	0	1	0	1	1	0	1	1	1	0	11	11
P10	0	1	1	1	0	1	1	0	1	0	1	1	0	1	1	0	0	10	10
P11	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	0	15	12
P12	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	0	15	14
P13	1	1	0	1	1	1	0	1	1	1	1	1	1	0	1	1	0	13	12
P14	1	1	0	0	1	1	0	1	0	1	1	1	1	0	1	1	0	11	10
P15	1	0	1	0	1	1	1	1	0	1	0	0	1	0	1	1	0	10	8
P16	1	0	1	1	1	1	1	1	1	1	0	0	1	0	0	1	0	11	8
P17	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	15	12
P18	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	15	12
P19	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	15	15
P20	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	0	14	13
P21	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	0	14	14
P22	1	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	11	11
P23	1	1	1	0	1	1	1	1	1	0	1	0	1	0	0	0	0	10	9
P24	1	1	1	1	1	1	1	1	0	1	1	1	1	0	0	0	0	12	12
P25	1	1	1	1	0	0	1	1	1	1	1	1	1	0	0	0	0	11	10
P26	1	1	1	1	1	1	1	0	1	1	1	1	1	0	0	0	0	12	9
P27	1	1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	0	11	9
P28	0	1	1	1	1	1	0	1	1	1	1	1	1	0	0	0	0	11	5
P29	1	0	1	0	0	1	0	1	1	0	0	0	1	0	0	0	0	6	6
P30	1	0	0	0	0	0	0	1	1	0	0	0	1	0	0	0	0	4	4
P31	1	0	0	0	0	0	0	1	1	0	0	0	1	0	0	0	0	4	4
P32	1	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	3	2

Table 4.3: Scheduling problem solution, with $w_1 = 1$, $w_2 = 0$ and $v_i = 1$ $i = 1, \ldots, 13$

Workforce scheduling model

rows - index of periods
1-13 - index of full time employees
I.,II.,III. - index of part time employees
EXTRA - number of extra employees in periods
AGENTS - number of agents in periods
NORA - number of required agents in periods

Cost of Schedule Distance from the optimal schedule
595 88

	1	2	3	4	5	6	7	8	9	10	11	12	13	I.	II.	III.	EXTRA	AGENTS	NORA
P1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	2	2
P2	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	2	2
P3	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	2	2
P4	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	3	3
P5	0	0	1	1	1	0	0	0	1	0	1	1	1	1	1	1	0	10	8
P6	0	1	1	1	1	0	0	0	1	0	1	1	1	1	1	1	0	11	11
P7	1	1	1	1	1	0	0	0	1	0	1	1	1	1	1	1	0	12	12
P8	1	1	1	1	1	1	0	1	1	0	1	1	1	1	1	1	0	14	13
P9	1	1	1	1	0	1	0	1	1	0	1	1	1	1	1	1	0	13	11
P10	1	1	1	1	0	1	0	0	1	0	1	0	1	1	1	0	0	10	10
P11	0	0	1	1	1	1	1	0	1	1	1	0	1	1	1	1	0	12	12
P12	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	14	14
P13	1	1	1	1	1	0	1	1	1	1	1	1	1	0	1	1	0	14	12
P14	1	1	1	0	1	0	1	1	1	1	1	1	0	0	1	1	0	12	10
P15	1	1	0	0	1	1	1	1	0	1	0	1	0	0	1	1	0	10	8
P16	1	1	0	1	1	1	1	1	0	1	0	1	1	0	0	1	0	11	8
P17	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	15	12
P18	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	15	12
P19	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	15	15
P20	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	15	13
P21	1	1	1	1	1	1	1	0	1	1	1	1	1	0	1	1	0	14	14
P22	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	13	11
P23	1	1	1	1	1	0	1	1	1	1	1	1	1	0	0	0	0	12	9
P24	1	1	1	1	1	1	1	1	0	1	1	1	1	0	0	0	0	12	12
P25	0	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	12	10
P26	1	0	1	0	1	1	1	1	1	1	1	1	1	0	0	0	0	11	9
P27	1	1	1	1	0	1	1	1	1	1	1	0	1	0	0	0	0	11	9
P28	1	1	0	1	1	1	0	1	1	0	0	1	0	0	0	0	0	8	5
P29	1	1	0	0	0	1	1	1	0	1	0	0	0	0	0	0	0	6	6
P30	1	0	0	0	0	1	1	1	0	1	0	0	0	0	0	0	0	5	4
P31	0	0	0	0	0	1	1	1	0	1	0	0	0	0	0	0	0	4	4
P32	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	2	2

Table 4.4: Scheduling problem solution, with $w_1 = 1$, $w_2 = 1$ and $v_i = 1$ $i = 1, \ldots, 13$

Workforce scheduling model

rows - index of periods
1-13 - index of full time employees
I.,II.,III. - index of part time employees
EXTRA - number of extra employees in periods
AGENTS - number of agents in periods
NORA - number of required agents in periods

Cost of Schedule	Distance from the optimal schedule
591	101

	1	2	3	4	5	6	7	8	9	10	11	12	13	I.	II.	III.	EXTRA	AGENTS	NORA
P1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	2	2
P2	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	2	2
P3	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	2	2
P4	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	3	3
P5	1	0	1	1	1	0	0	0	1	0	1	1	0	1	1	1	0	10	8
P6	1	0	1	1	1	1	0	0	1	0	1	1	1	1	1	1	0	12	11
P7	1	0	1	1	1	1	0	0	1	0	1	1	1	1	1	1	0	12	12
P8	1	0	1	1	1	1	0	1	1	1	1	1	1	1	1	1	0	14	13
P9	1	0	1	1	0	1	0	1	1	1	0	1	1	1	1	1	0	12	11
P10	1	0	1	1	0	1	0	1	1	1	0	0	1	1	1	0	0	10	10
P11	1	1	1	0	1	1	1	1	1	1	1	0	1	0	1	1	0	13	12
P12	1	1	1	0	1	1	1	1	1	1	1	1	1	0	1	1	0	14	14
P13	1	1	1	1	1	1	1	0	1	0	1	1	1	0	1	1	0	13	12
P14	0	1	1	1	1	1	1	0	1	0	1	1	1	0	1	1	0	12	10
P15	0	1	0	1	1	0	1	1	0	1	1	1	0	0	1	1	0	10	8
P16	1	1	0	1	1	0	1	1	0	1	1	1	0	0	0	1	0	10	8
P17	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	15	12
P18	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	15	12
P19	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	15	15
P20	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	0	0	14	13
P21	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	0	0	14	14
P22	1	1	0	1	1	0	1	1	1	1	1	1	1	0	0	0	0	11	11
P23	1	1	1	1	1	1	1	0	0	1	1	1	1	0	0	0	0	11	9
P24	1	0	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	12	12
P25	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	13	10
P26	0	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	11	9
P27	1	1	1	1	1	1	0	1	1	1	1	1	1	0	0	0	0	12	9
P28	1	1	1	0	0	1	1	1	1	0	0	0	1	0	0	0	0	8	5
P29	0	1	0	0	0	1	1	1	0	1	0	0	1	0	0	0	0	6	6
P30	0	1	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	4	4
P31	0	1	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	4	4
P32	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	2	2

Table 4.5: Scheduling problem solution, with $w_1 = 1$, $w_2 = 1$, $v_i = 1 \; \forall \; i \neq 1, 2, 3$, $v_1 = 2$, $v_2 = 2$ and $v_3 = 2$

Workforce scheduling model

rows - index of periods
1-13 - index of full time employees
I.,II.,III. - index of part time employees
EXTRA - number of extra employees in periods
AGENTS - number of agents in periods
NORA - number of required agents in periods

Cost of Schedule Distance from the optimal schedule
593 74

	1	2	3	4	5	6	7	8	9	10	11	12	13	I.	II.	III.	EXTRA	AGENTS	NORA
P1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	2	2
P2	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	2	2
P3	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	2	2
P4	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	3	3
P5	1	0	0	1	1	0	0	0	0	0	1	1	1	1	1	1	0	9	8
P6	1	0	1	1	1	1	0	0	1	0	1	1	1	1	1	1	0	12	11
P7	1	0	1	1	1	1	0	0	1	0	1	1	1	1	1	1	0	12	12
P8	1	1	1	1	1	1	0	0	1	0	1	1	1	1	1	1	0	13	13
P9	1	0	1	1	1	1	0	0	1	0	1	1	1	1	1	1	0	12	11
P10	0	0	1	1	1	1	0	0	1	0	1	1	1	1	1	0	0	10	10
P11	0	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	0	14	12
P12	1	1	1	1	1	0	1	1	1	1	1	1	0	1	1	1	0	14	14
P13	1	1	1	1	0	1	1	1	1	1	1	1	0	0	1	1	0	13	12
P14	1	1	1	0	0	1	1	1	1	1	1	0	1	0	1	1	0	12	10
P15	1	1	0	0	1	1	1	1	0	1	0	0	1	0	1	1	0	10	8
P16	1	1	0	1	1	1	1	1	0	1	0	1	1	0	0	1	0	11	8
P17	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	15	12
P18	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	15	12
P19	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	15	15
P20	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	0	14	13
P21	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	0	14	14
P22	1	1	1	1	1	0	1	1	1	1	1	1	1	0	0	0	0	12	11
P23	1	1	1	0	1	1	1	1	1	1	1	1	1	0	0	0	0	12	9
P24	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	13	12
P25	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	13	10
P26	1	0	1	1	1	1	1	1	0	1	1	1	0	0	0	0	0	10	9
P27	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	13	9
P28	0	1	0	1	0	1	0	0	1	0	0	0	1	0	0	0	0	5	5
P29	0	1	1	0	0	1	1	1	1	1	0	0	0	0	0	0	0	7	6
P30	0	1	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	4	4
P31	0	1	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	4	4
P32	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	3	2

Table 4.6: Scheduling problem solution, with $w_1 = 1$, $w_2 = 5$ and $v_i = 1$ $i = 1, \ldots, 13$

Workforce scheduling model

rows - index of periods
1-13 - index of full time employees
I.,II.,III. - index of part time employees
EXTRA - number of extra employees in periods
AGENTS - number of agents in periods
NORA - number of required agents in periods

Cost of Schedule	Distance from the optimal schedule
595	97

	1	2	3	4	5	6	7	8	9	10	11	12	13	I.	II.	III.	EXTRA	AGENTS	NORA
P1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	2	2
P2	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	2	2
P3	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	2	2
P4	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	3	3
P5	1	0	0	1	1	0	0	0	1	0	1	1	1	1	1	1	0	10	8
P6	1	0	0	1	1	1	0	0	1	0	1	1	1	1	1	1	0	11	11
P7	1	0	1	1	1	1	0	0	1	0	1	1	1	1	1	1	0	12	12
P8	1	0	1	1	1	1	0	0	1	1	1	1	1	1	1	1	0	13	13
P9	1	0	1	1	1	1	0	0	1	1	1	1	1	1	1	1	0	13	11
P10	1	0	1	1	1	1	0	0	1	1	1	0	1	1	1	0	0	11	10
P11	1	1	1	1	1	1	1	1	1	1	0	0	1	1	1	1	0	14	12
P12	1	1	1	1	1	1	1	1	1	1	0	1	0	1	1	1	0	14	14
P13	1	1	1	1	1	1	1	1	0	0	1	1	0	1	1	1	0	13	12
P14	1	1	1	0	0	1	1	1	0	0	1	1	1	1	1	1	0	12	10
P15	0	1	0	0	0	0	1	1	1	1	1	1	1	0	1	1	0	10	8
P16	0	1	0	1	1	0	1	1	1	1	1	1	1	0	0	1	0	11	8
P17	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	15	12
P18	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	15	12
P19	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	15	15
P20	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	0	14	13
P21	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	0	14	14
P22	1	1	1	0	1	1	1	1	0	1	1	1	1	0	0	0	0	11	11
P23	1	0	0	1	1	0	1	1	1	1	1	1	1	0	0	0	0	10	9
P24	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	13	12
P25	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	13	10
P26	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	12	9
P27	0	1	1	1	0	1	1	1	1	1	1	1	1	0	0	0	0	11	9
P28	1	1	1	1	1	1	0	0	1	0	0	0	1	0	0	0	0	8	5
P29	0	1	1	0	0	1	1	1	0	1	0	0	0	0	0	0	0	6	6
P30	0	1	1	0	0	0	1	1	0	1	0	0	0	0	0	0	0	5	4
P31	0	1	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	4	4
P32	0	1	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	3	2

Table 4.7: Scheduling problem solution, with $w_1 = 1$, $w_2 = 5$, $v_i = 1 \ \forall \ i \neq 1, 2, 3$, $v_1 = 2$, $v_2 = 2$ and $v_3 = 2$

Chapter 5

Concluding remarks and future prospects

The main interest of this thesis was to introduce and apply a mathematical programming model for labor scheduling, which has the following properties:

1. it models implicitly shifts with logical constraints,
2. it solves the shift scheduling and the employee assignment problem together,
3. it models employees' preferences.

In Chapter 2 we built up this model and introduced methods for express logical relationships between variables. We used several methods out of these during our experiments to express specific conditions of the problem. We have already shown that it is possible to express the same condition in different ways due to the equivalences, which are shown in Section 2.2. This gives us an opportunity for the future to find the most proper way to express different kinds of relationships in models and thus decrease the number of variables and constraints in the model.
We defined a metric in Subsection 2.4.3 and used it to involve employees' preferences to the model. If we study the properties of this metric we can see that our choice was rational. Of course it does not mean that this is the only way we can express these preferences. We can use different metrics or completely different expressions instead of our metric. Fortunately, the change of this expression does not have big effect on the structure of our modeling method, i.e., on the logical constraints. Even so, it is possible that we use an expression, which is complicated to implement in an LP or IP environment. We described our method, with which we expressed the metric, in Subsection 4.3.1.

Our experimental results depends on both the implementation of the problem and the IP solver what we use. It can be very useful if we use other modeling techniques to solve the problem. Using different logical implementations, different modeling techniques and different solvers enables us to compare and analyze our results. This comparison and analysis can be the next step in the future.

Bibliography

[1] Aykin, T. Optimal shift scheduling with multiple break windows. *Management Science vol. 42* (1996), pp. 591–602.

[2] Bechtold, S. E., and Jacobs, L. W. The equivalence of set-covering and implicit optimal integer programming formulations for mathematical labor scheduling problems. Tech. rep., Florida State University, 1989.

[3] Bechtold, S. E., and Jacobs, L. W. Implicit modelling of flexible break assignments in optimal shift scheduling. *Management Science vol. 36* (1990), pp. 1339–1351.

[4] Dantzig, G. B. A comment on Edie's traffic delays at toll booth. *Operations Research vol. 2(3)* (1954), pp. 339–341.

[5] Edie, L. C. Traffic delays at toll booths. *Operations Research vol. 2* (1954), pp. 107–138.

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

[7] Koole, G., and Mandelbaum, A. Queueing models of call centers. *Annals of Operations Research vol. 113* (2002), pp. 41–59.

[8] Thompson, G. M. Assigning telephone operators to shifts at New Brunswick Telephone Company. *Interfaces vol. 27* (1997), pp. 1–11.

[9] Williams, H. P. *Model Building in Mathematical Programming.* Chichester Wiley, 1998.