Problem Model for Nurse Rostering Benchmark Instances

Edmund K. Burke[1], Tim Curtois[1], Rong Qu[1], and Greet Vanden-Berghe[2]

[1] ASAP, School of Computer Science,
University of Nottingham, Jubilee Campus, Nottingham, UK
[2] KaHo St.-Lieven Gebr. Desmetstraat 1 9000 Gent, Belgium

Abstract. This document describes the model used in the nurse rostering data sets found at http://www.cs.nott.ac.uk/˜tec/NRP/ The model is based on and very similar to ANROM (Advanced Nurse Rostering Model) [1].

1 Plane, nurse rostering system for Belgian hospitals

ANROM was created during the development of the automated rostering module of Plane. Plane is a commercial software system developed in a collaboration between Impakt N.V. and GET for assistance in hospital employee scheduling. Plane arose from the need for automated rostering assistance in Belgian healthcare organisations. In such a rostering problem, personnel requirements for every skill category have to be fulfilled over a fixed period in time, while respecting a number of constraints that limit the personal assignments.
The initial version of Plane was first implemented in a hospital in 1995 but the system evolved to cope with the new and more complex real-world problems that keep appearing. Over 40 hospitals in Belgium, of which some have about 100 wards, replaced their time consuming manual scheduling with this system.
Although the problem is user-defined to a large extent, the software has to be efficient in different settings. Every specific hospital ward should be able to formulate its problem within the restrictions of the model described in the following sections.

2 Dimensions

2.1 Personnel

Hospitals are organised in wards with fixed activities, usually a settled location, and, for the most part, a permanent team of nurses. Although practical situations often allow people to migrate to another ward whenever a personnel shortage is unsolvable, the personnel rostering problem in this research normally concerns a group of personnel belonging to the same ward. Planning the different hospital wards reduces the complexity of the problem considerably.

In the personnel rostering model of this research, the number of personnel members is user definable. It is not the result of calculations within the planning algorithm. Staffing considerations and decisions on the capacity in terms of beds and patients, are beyond the competence of the short-term timetablers in Belgian hospitals. The number of people P in a ward varies in practice from about 20 to over 100.

2.2 Skill categories

Personnel members in a ward belong to skill categories. The division into categories is based upon the particular level of qualification, responsibility, job description, and experience of the personnel. Typical categories in hospitals are: *head nurse*, *regular nurse*, *junior nurse*, *ambulance driver*, *caretaker*, *cleaner*, etc. The groups are called 'grades' in some other applications. In our model, there are Q different skill categories. Each personnel member p ($1 \leq p \leq P$) belongs to one skill category q_p.
The approach in ANROM allows for a personalised organisation of substitution among skill categories. Rather than strictly disjunctive skill categories or hierarchical substitutability (in which higher skilled people can replace less experienced colleagues), we opted for a solution that is closer to the reality in hospitals. For example, a particular skill category might be a class of junior nurses. It might be the case that we could allocate someone in the ward manager's category to a junior shift on a given day. In practice, very senior staff are usually reluctant to stand in for junior staff. It is also the case that, in practice, a regular (not a junior) nurse will temporarily stand in for a head nurse.
The problem of replacing people is solved in ANROM by assigning *alternative* skill categories to certain people. People with more experience or who have taken some exams, can be substitutes for higher skill categories. The nurses who might replace a head nurse, for example, normally are a couple of senior nurses who know everything about the working of the ward. We denote by QA_p the list of alternative skill categories for person p. The number of elements in the list is 0 when the person p has no permission to replace people from other skill categories than q_p and QA_p contains $Q-1$ elements if p can do the work of any other skill category in the ward.

2.3 Work regulations

Cyclical schedules obey very strict patterns. They are applied in many personnel rostering environments but are very impractical in real healthcare environments. In our model, hospital personnel have work regulations or contracts with their employer. Most organisations allow several job descriptions such as part time work, night nurses' contracts, weekend workers, etc. These regulations involve different constraints but they can make the schedules much more flexible. Moreover, very personal arrangements like 'free Wednesday afternoons' or 'refresher courses' at regular points in time, etc can easily be formulated. It is not unlikely to have personalised contracts for the majority of personnel members in Belgian

hospitals.
The different work regulations are denoted by w ($1 \leq w \leq W$) in this research. For every personnel member p, the work regulation is denoted by w_p. .

2.4 Shift types

A shift type is a predefined period with a fixed start and end time in which personnel members can be on or off duty. Many continuously working organisations schedule three typical shift types called *morning*, *late* and *night shift*. This is the way that manufacturing generally works. However, these working hours cannot cover the personnel requirements of hospitals in practice. Moreover, all the possible kinds of part time work require a variety in start and end times and in shift length. There are S shift types per ward. Each shift type s has a start time $shift\_start_s$, an end time $shift\_end_s$, and a duration $shift\_duration_s$. Table 1 presents a simplified example of a set of shift types in a ward. A shift type

		Start	End
M	morning shift	06:45	14:45
L	late shift	14:30	22:00
N	night shift	22:00	07:00

Table 1. Example of shift types

does not always involve continuous activity between the start and the end time, and hence $shift\_duration_s$ is not necessarily equal to $shift\_end_s - shift\_start_s$. Examples of such shift types are those in which a long break enables personnel to have lunch at home, or guard duties, which require availability from the personnel during a longer period than the actual time which is counted as working time.

2.5 Planning period

Planning periods for nurse rostering vary from a couple of days to a few months. The length of the period is expressed as a number of days D, or a number of assignment units T (explained in Section 2.6). Since cyclical rosters are not common at all, it is important for individual employees to know their schedule some time in advance. Long term scheduling, on the other hand, should not be too detailed because the personnel demand and personal preferences fluctuate and are not predictable but for the near future. In this model, we always consider planning periods which start on Monday and end on Sunday, no matter what the duration (number of weeks) is.
Short planning periods enable the search algorithms to find good quality results much faster than longer planning periods. However, very short planning periods

reduce the possibility of guaranteeing fairness among personnel members. Some situations require shorter planning periods. Examples are unexpected changes in the requirements or the constraints. In such cases, the personnel manager tends to prefer the fewest modifications possible in the people's schedules.
ANROM provides some planning procedures for organising rescheduling processes. Parts of the already existing roster can be 'frozen' during the planning. Both personal schedules and periods in time (for all the members of the ward) can be kept unchanged while the algorithms search for a better solution in the remaining part of the problem domain.

2.6 Schedule

The roster of the ward, in which the shift assignments to people are stored, is called the *schedule*. It has dimensions P and T, where T stands for the number of *assignment units*. We will refer to a personal schedule for person p by the notation $schedule_p$ and to a particular assignment at unit t by the notation $schedule_{p,t}$.
We define assignment units as entities of minimum allocation in a schedule. They are mainly introduced to handle the soft constraints on the personnel's schedules. In the approach of ANROM, where personnel requirements and schedules make use of shift types, each shift type on each day has a corresponding assignment unit. The total number of different assignment units T for a schedule is therefore equal to $S * D$. An assignment unit t_s $(1 \leq s \leq S)$ corresponds to shift type s. For the clarity of notations, s_t is introduced as the shift type that corresponds to the assignment unit t.
We illustrate the meaning of assignment units with a simple example. A fragment of a possible personnel roster is presented in Fig. 1. We notice that there are five people in the ward, and that there are three different shift types. Fig. 2 presents the *schedule* that corresponds to the roster of Fig. 1. Each column in the schedule represents an assignment unit. For every day of the planning period there are S columns, each of them corresponding to a shift type. The assignment

	Mon	Tue	Wed	Thu	Fri	Sat	Sun
P1	M	M	L	L	N		
P2	N		N	L	L		
P3	M	M	M	M	M	M	M
P4	M		L	N	N	N	
P5	M L	L	L	L			

Fig. 1. Roster example for 5 people (P1,..., P5) and 1 week; M, L, and N being the shift types introduced in Table 1

units are ordered according to the start times of the shift types they represent. When two shift types have the same start time, the first assignment unit will

	Schedule Example																					
P1	*	-	-	*	-	-	-	*	-	-	*	-	-	-	*	-	-	-	-	-	-	
P2	-	-	*	-	-	-	-	-	*	-	*	-	-	*	-	-	-	-	-	-	-	
P3	*	-	-	*	-	-	*	-	-	*	-	-	*	-	-	*	-	-	*	-	-	
P4	*	-	-	-	-	-	-	*	-	-	-	*	-	-	*	-	-	*	-	-	-	
P5	*	*	-	-	*	-	-	*	-	-	*	-	-	-	-	-	-	-	-	-	-	

Fig. 2. Schedule corresponding to the roster in Fig. 1: '*' denotes that there is an assignment in the schedule, '-' denotes that the schedule is free, * will be specified in Section 3

match the shift type with the earliest end time. The assignment units defined for this approach do not represent consecutive or separate periods but they will very often overlap in time. The number of assignment units equals the number of shift types times the number of days in the planning period ($T = S * D$, where D denotes the number of days in the planning period).

3 Hard Constraints

Hard constraints are those that must always be satisfied. We can cover most real-world hospital situations with the hard constraints of the following set.

3.1 Maximum one assignment per shift type per day

In ANROM, it is not considered feasible to assign the same shift to a member of the ward more than once per day. A violation of this constraint would involve twice the workload for that person. The constraint can be represented by (1), in which *pref* is a number indicating that assignments have been made in a post-planning coverage procedure. A value 0 at a schedule position $schedule_{p,t}$ indicates that there is no assignment for person p at assignment unit t. Any value different from 0 denotes the skill category for which an assignment is made. It is only in exceptional cases that a person will have an assignment for another than his or her main skill category. The values $pref+1$ to $pref+Q$ represent the skill categories 1 to Q but the added number *pref* indicates how the assignments were made.

$$\forall p, (1 \leq p \leq P);\ \forall t, (1 \leq t \leq T): \\ schedule_{p,t} \in \{0, 1, \ldots, Q, pref+1, \ldots, pref+Q\} \quad (1)$$

Within a ward, the schedule representation of ANROM thus prevents personnel members from having more than one assignment per shift and per day, be it for their main or for an alternative skill category. It is not the case that this constraint prevents people from being assigned to overlapping shifts, a condition that will be handled by a soft constraint in Section 4.

3.2 Required skill

In case the number of people required for a skill category is higher than the available number, those who have the skill as an alternative grade can assist (see also Constraint 2 in Section 4). It is infeasible to assign tasks to people who are not qualified to carry them out. This is formally presented in (2), in which the values that can occur in a personal schedule are restricted to the following set.

$$\forall p, (1 \leq p \leq P);\ \forall t, (1 \leq t \leq T) :$$
$$schedule_{p,t} \in \{0, q_p, q_p + pref\} \cup QA_p \cup QA_p + pref \quad (2)$$
$$with \quad A + a = \{x\ \xi\ (x - a) \in A\}$$

3.3 Personnel requirements

Personnel requirements, also called 'coverage constraints', express the number of personnel of each skill category needed to staff the ward. They are set by management and are usually expressed in terms of the minimum number of personnel required and the preferred number of personnel to meet patients' needs.

4 Soft Constraints

4.1 Introduction

The real-world situation addressed in this research incorporates a high number of soft constraints on the personal schedules. The soft constraints will preferably be satisfied, but violations can be accepted to a certain extent. It is highly exceptional in practice to find a schedule that satisfies all the soft constraints. The aim of the search algorithms is to minimise the real impact of violations of these constraints. The users of the system specify all the constraints.
Some of the constraints strengthen other constraints, while others are adverse factors in the planning of real-world hospital wards. Very often a few constraints are even contradictory in reality. It is sometimes obvious that certain constraints can never be satisfied at all. In all of these cases, the user of the planning software must be informed about the extent to which each type of constraint is violated.

4.2 Relaxation of constraints

Apart from describing the meaning of every constraint, we also explain some exceptions for the evaluation in addition to certain corrections which are required in holiday periods or periods of illness absence. Boundary constraints at the beginning and end of the planning period have an important impact on the evaluation. In general, the rule holds that a penalty is generated when a violation of a constraint could be avoided in the current planning period by scheduling

appropriately. No violation is generated when the constraint, which is not satisfied, can still be satisfied by scheduling appropriate shifts in the next planning period.
The evaluation of an absence and a free day can be very different. Absences are personal constraints such as holidays, illness, etc (Constraint 22 and 23 in Section 4.3). Free days are days in a personal schedule on which nothing is scheduled. Manual planners evaluate some constraints in a less strict manner when they are violated by a *day off* than in the case where the violation is caused by a free day. Some constraints are relaxed when an absence prevents scheduling shifts which could satisfy the constraint. Free days cannot allow for a compensation because free days can become scheduled days by assigning a shift. The same ideas hold when distinguishing between scheduled shifts and requested shifts for some particular soft constraint types (Constraint 24), as explained in Section 4.3.

4.3 Categories of soft constraints

Hospital constraints Personnel scheduling is organised per hospital ward in this research project. A ward is a group of personnel working together in the same location (e.g. a certain floor in a hospital) or on the basis of their activities (e.g. the ambulance team). A number of general constraints are recommended by the hospital but in certain situations, they may need to be ignored. Next to the rules which hold in the entire hospital, each ward can define their house rules. The general hospital constraints will be listed in turn and explained in detail.

Constraint 1 *Minimum time between two assignments*

There is a legal constraint depicting how many hours personnel should be free between two assignments. In practice, the time between two assignments depends on the shift types. The formulation of this constraint is represented by two extra data fields corresponding to the shift types. In ANROM, planners can decide to augment or diminish the rest time before and after each shift type. Scheduling a morning shift which starts only 7 hours after a late shift has ended is obviously worse than scheduling a short afternoon shift within 7 hours after a short morning shift, for example. Before and after very short duties (like a morning shift from 8 till 12) a shorter break can be acceptable. The terms $shift\_before_s$ and $shift\_after_s$ denote the recommended free time before shift type s starts and after it ends. A penalty will be generated whenever there is an overlapping in time between a shift type and a forbidden zone from another shift. Since this constraint is always very important, the penalty is proportional to the number of overlapping minutes. Table 2 presents an example, extracted from a real-world hospital situation, in which the start and end times of the shift types, the recommended idle times before and after them are shown. The names of the shift types are abbreviations of the shift names in the hospital. Fig. 3, which is derived from the data in Table 2, illustrates the constraint better. Each shift type is presented as a bar in the figure. The start and end time of the shifts

are clearly visible and forbidden sequences between shift types can be derived from the periods before and after the shift, in which no other work is allowed. The constraint restricts shift types scheduled on a certain day, in addition to assignments on consecutive days in some particular cases. It makes no sense to check this constraint for assignments that are two days or more apart. The time between them will always suffice. The penalty for Constraint 1 is denoted by $penalty_{p,C1}$ for person p. The formal representation is given in (3).

$$
\begin{aligned}
&\forall p, (1 \leq p \leq P),\ \forall t_1, (1 \leq t_1 \leq T),\ \forall t_2, (t_1 < t_2 \leq T) : \\
&\quad penalty_{p,C1} = 0 \\
&\quad s_1 = s_{t_1}, s_2 = s_{t_2} \\
&\quad d_1 = t_1/S, d_2 = t_2/S \\
&\quad x = shift\_end_{s_1} + shift\_after_{s_1} - (shift\_start_{s_2} + 24 * 60 * (d_2 - d_1)) \\
&\quad y = shift\_end_{s_1} - (shift\_start_{s_2} + 24 * 60 * (d_2 - d_1) - shift\_before_{s_2}) \\
&\quad IF\ (d_1 + 2 \geq d_2 \wedge schedule_{p,t_1} \neq 0 \wedge schedule_{p,t_2} \neq 0) \\
&\qquad \Rightarrow penalty_{p,C1} + \begin{cases} x\ IF\ (x > 0) \\ y\ IF\ (y > 0) \end{cases}
\end{aligned}
\tag{3}
$$

				Unoccupied Time (hrs)	
Shifts		$shift\_start_s$	$shift\_end_s$	$shift\_before_s$	$shift\_after_s$
SE	'Short' Early	6:00	10:00	10	8
EE	'Early' Early	6:00	14:00	10	8
SD	'Short' Day	8:00	12:00	8	8
E	Early	8:00	17:00	8	8
D	Day	10:00	18:00	8	8
SL	'Short' Late	14:00	18:00	8	8
L	Late	14:00	22:00	8	10
LL	'Late' Late	17:00	22:00	8	10
N	Night	22:00	6:00	10	10

Table 2. Start and end times of a realistic shift type set; unoccupied periods before and after

Constraint 2 *Alternative skill category*

It is a hard constraint that all the work has to be done by skilled personnel. If a certain duty requires a head nurse, then preferably a head nurse will do the job unless there is none available. The only possibility to still obtain a feasible solution in that case is to find a person from another skill category who is

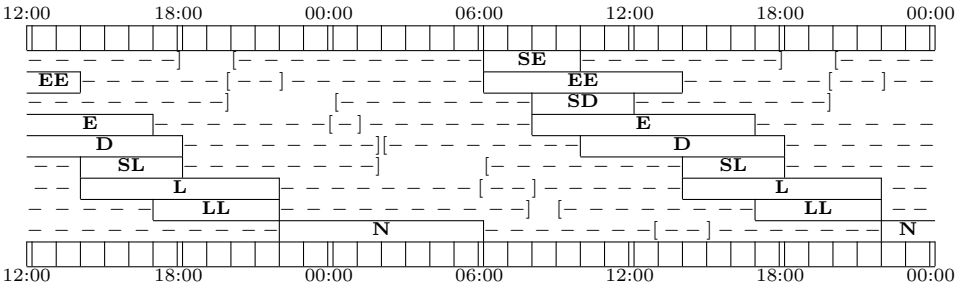

Fig. 3. Minimum time between two shift types

authorised to replace people from the required skill category (see Section 2.2). Assigning people from alternative skill categories is sometimes necessary to cater for staff shortages, but it is not desirable and will be penalised in the evaluation function. A penalty is generated each time a shift is performed for a duty other than ones that are covered by the prime skill category: $penalty_{p,C2}$. This is presented in (4).

$$\forall p, (1 \leq p \leq P): \quad penalty_{p,C2} = |\{t\ \xi\ 1 \leq t \leq T \wedge schedule_{p,t} \in QA_p \cup QA_p + pref\}| \quad (4)$$

Constraints defined by the work regulation Every personnel member has a contract with the hospital. It is called the work regulation or work agreement (see Section 2.3). There are different work regulations for full time and half time personnel members, night nurses, etc. Many hospitals in Belgium allow for the definition of a personal work agreement per nurse. This enables them to formulate personal constraints such as *every Wednesday afternoon free*, *work a weekend every two weeks*, *no stand by duty*, etc. When defining the work regulation, either of the following constraints can be defined or made idle.

Constraint 3 *Maximum number of assignments*

This constraint determines the number of shifts a person with the work regulation w can -at most- work during the planning period: max_w. In order to reflect the real-world situation, adaptations to this number are made in cases of illness, holiday, ... When a personal schedule contains an 'absence', the type of leave or absence is given. Depending on the reason for the absence, it will be taken into account for the evaluation of the assignments and hours. We denote by A_p the number of absence days within the planning period. Consider a planning period of 4 weeks (28 days) and a *full time* work agreement with maximum 20 assignments during the planning period. Suppose a full time personnel member is on sickness leave during 10 days. The constraint is adapted to this situation by changing the value of the maximum number of assignments from

20 to $\lceil 20 * (28 - 10)/28 \rceil$. The cost function value for the constraint, $penalty_{p,C3}$ for person p, is calculated per planning period. There is no transfer of excesses to the next planning period. This is represented by (5).

$$\begin{aligned} &\forall p, (1 \leq p \leq P): \\ &\qquad penalty_{p,C3} = x \ IF \ (x > 0) \\ &\qquad w = w_p \\ &\qquad x = |\{t \ \xi \ 1 \leq t \leq T \wedge schedule_{p,t} \neq 0\}| - \tfrac{D - A_p}{D} * max_w \end{aligned} \tag{5}$$

Constraint 4 *Maximum number of consecutive days*

This constraint limits the maximum number of consecutive working days by $max\_consecutive\_days_w$. The evaluation involves checking the days which are scheduled at the end of the previous planning period ($previous\_consecutive\_days_p$). In practice, the schedule of the previous planning period is known and the consecutive days at the end of it can be calculated. For the description of the soft constraints in this section, we assume that the required data is given. Suppose, for example, that the maximum number of consecutive days is 7. If 6 consecutive days are scheduled at the end of the previous planning period, the first day of the current planning period can be scheduled without violating the constraint. Scheduling both the first and the second day will generate a penalty. When the last 8 consecutive days at the end of the previous planning period have assignments, no penalty will be generated in the current period when there is nothing scheduled on the first day. Constraint violations made in the past are not penalised in the current planning period. This is formally presented in (6).

Constraint 5 *Minimum number of consecutive days*

A working day between two free days is seldom wanted. We denote $min\_consecutive\_days_w$ as the minimum number of consecutive working days for the work regulation w. Also for this constraint, the previous planning period for person p is taken into account by $previous\_consecutive\_days_p$. A penalty will be generated if the constraint could be satisfied by scheduling a duty on the first day(s) of the current planning period. Consequently, it is not considered a violation when the constraint is not satisfied at the end of the planning period. If, for example, at least 3 consecutive days are required, no penalty will be generated if only the last two days of the current planning period are scheduled. The constraint on the minimum number of consecutive days is relaxed when a stretch of working days and requested days off meets the requirement. Consider the case where at least three consecutive days are required. Suppose a person works two consecutive days and ends with a requested day off immediately afterwards. This schedule will not

$$
\begin{array}{l}
\forall p, (1 \leq p \leq P): \\
\qquad w = w_p \\
\qquad consecutive\_days = previous\_consecutive\_days_p \\
\qquad penalty_{p,C4} = 0 \\
\\
\qquad \forall d, (1 \leq d \leq D): \\
\\
\qquad \left\{ \begin{array}{ll}
x = |\{t \ \xi \ 1 \leq t \leq S \wedge schedule_{p,(d-1)*S+t} \neq 0\}| & \\
 & \\
consecutive\_days + 1 & IF \quad (x \geq 1) \\
\left. \begin{array}{l} y = consecutive\_days - max\_consecutive\_days_w \\ penalty_{p,C4} + y, \ IF \quad (y > 0) \\ consecutive\_days = 0 \end{array} \right\} & IF \quad (x = 0)
\end{array} \right.
\end{array}
\tag{6}
$$

generate a penalty because the requested day off is, unlike free days, considered part of the work stretch. A day off d, for person p is denoted by 1 in $days\_off_{p,d}$ (see also Constraint 22) other days are denoted by 0. This is illustrated by (7).

Constraint 6 *Maximum number of consecutive free days*

The value $max\_cons\_free\_days_w$ denotes the maximum number of consecutive free days for work regulation w. The number of consecutive free days at the end of the previous planning period is denoted by $previous\_consecutive\_free\_days_p$ for person p. The constraint is presented by (8).

Constraint 7 *Minimum number of consecutive free days*

The minimum number of consecutive free days for work regulation w is denoted by $min\_cons\_free\_days_w$. More formal detail is given in (9). Constraint 6 and 7 are analogous to the previous two constraints, they limit the consecutive free days instead of the consecutive working days. The same rules hold at the beginning and the end of the planning period. With respect to absences and free days, the most relaxed attitude holds. Absence days are not added to the number of consecutive free days (even if they occur in the sequence) when the maximum number is evaluated. Absence days are added to the number of consecutive free days for the minimum constraint. There is no violation for the minimum number of consecutive free days, when absence days are isolated (flanked by working days).

Constraint 8 *Maximum number of hours worked*

The limit on the maximum number of hours during a planning period is given by max_hours_w for the corresponding work regulation. Unlike most other constraints, the working time is cumulative and this also affects the evaluation of Constraint 9. By adding the real amount of overtime or undertime to the scheduled time in the next planning period, the system prevents the team from

$$
\begin{array}{l}
\forall p, (1 \leq p \leq P): \\
\quad w = w_p \\
\quad consecutive\_days = previous\_consecutive\_days_p \\
\quad consecutive\_days\_off = previous\_consecutive\_days\_off_p \\
\quad penalty_{p,C5} = 0 \\
\\
\quad \forall d, (1 \leq d \leq D): \\
\\
\quad \left\{ \begin{array}{ll}
x = |\{t \; \xi \; 1 \leq t \leq S \wedge schedule_{p,(d-1)*S+t} \neq 0\}| & \\
consecutive\_days + 1, & IF \;\; (x \geq 1) \\
cons\_days\_off + 1, & IF \;\; (x = 0 \wedge days\_off_{p,d} = 1) \\
\\
\left. \begin{array}{l}
y = min\_consecutive\_days_w - consecutive\_days \\
\qquad -cons\_days\_off \\
penalty_{p,C5} + y, \qquad IF \;\; (y > 0) \\
consecutive\_days = 0 \\
cons\_days\_off = 0
\end{array} \right\} & \\
\qquad\qquad\qquad\qquad\qquad\qquad IF \;\; (x = 0 \wedge days\_off_{p,d} = 0) &
\end{array} \right.
\end{array}
\tag{7}
$$

having unfair schedules. The balance of the working hours for personnel member p, which is transferred to the next planning period, ($previous\_hours_p$) is added to the hours carried out in the current planning period. Suppose the starting balance for a person is negative in the current planning period. There is a possibility for compensating undertime without generating a penalty for overtime. Overtime is very common in hospitals, so an option is available for not penalising overtime unless a certain threshold $threshold\_hours$ is exceeded. The value is the same for all the personnel in the ward. When violations are less than this given number of hours, the penalty is ignored.
Every work regulation has a $standard\_performance_w$, which gives the normal number of hours worked on a standard day. We call $AH_p = standard\_performance_w * A_p$ the number of hours on absence days. In nearly the same way as explained with respect to Constraint 3, a weighted new value for the maximum number of hours is calculated. This is demonstrated in (10).

Constraint 9 *Minimum number of hours worked*

The minimum number of hours a person should work during the planning period is min_hours_w. Evaluating the constraint is carried out in the same way as Constraint 8. The balance of hours worked at the end of the previous planning period is added to the hours of the current period. Absence days or illness can be compensated in the evaluation of the constraint. The constraint is illustrated in (11). The evaluation of the constraint can be relaxed by setting a threshold value for generating penalties. When violations are less than a given number of hours, it is possible to ignore these violations of the constraint. Since the real amount of overtime or undertime will be added to the scheduled time in the next

$$
\begin{array}{l}
\forall p, (1 \leq p \leq P): \\
\qquad w = w_p \\
\qquad cons\_free\_days = previous\_consecutive\_free\_days_p \\
\qquad penalty_{p,C6} = 0 \\
\\
\qquad \forall d, (1 \leq d \leq D): \\
\\
\qquad \left\{ \begin{array}{ll}
x = |\{t \ \xi \ 1 \leq t \leq S \wedge schedule_{p,(d-1)*S+t} \neq 0\}| & \\
consecutive\_free\_days + 1 & IF \ (x = 0) \\
\left. \begin{array}{l} y = cons\_free\_days - max\_cons\_free\_days_w \\ penalty_{p,C6} + y, \ IF \ (y > 0) \end{array} \right\} & IF \ (x \geq 1) \\
cons\_free\_days = 0 &
\end{array} \right.
\end{array}
\tag{8}
$$

planning period, the system prevents the team from having unfair schedules. A correction is calculated when a person is absent due to illness or a holiday. In exactly the same way as explained with respect to Constraint 3, a weighted new value for the maximum and minimum number of hours is calculated.

Constraint 10 *Maximum number of assignments per day of the week*

This constraint limits the number of assignments on certain days of the week by $max\_day_{w,day}$, in which w denotes the work regulation and *day* is the day of the week (Monday till Sunday). It is, for example, possible to provide at least one free Monday during the planning period or to restrict the number of working weekends with this constraint. There is no transfer between planning periods. The constraint is formally presented in (12).

Constraint 11 *Maximum number of assignments for each shift type*

This constraint provides the possibility of forbidding and/or restricting the assignment of certain shift types *shift* for the work agreement by $max\_shift_{w,shift}$. The planner can, when for example defining a work agreement for night nurses, set to 0 the number of allowed shift types for every shift that differs from the night shift. Other work agreements, like cleaner for example, will never work a night shift. Very often, the maximum number for each shift type is set to a rather low number in order to enable shift type variation in the schedules. The constraint is demonstrated formally in (13).

Constraint 12 *Maximum number of a shift type per week*

For every week *week* in the planning period, the user can limit the number of assignments in a personal schedule for every shift type *shift* by $max\_shift\_week_{w,shift,week}$. This constraint can, for example, prevent the assignment of seven night duties in one week. Since the system allows different constraint values for different weeks, it can also allow for the definition of shift type cycles like one 'early week followed by a late week'. It can be formally illustrated as in (14).

$$
\begin{array}{l}
\forall p, (1 \leq p \leq P): \\
\quad w = w_p \\
\quad cons\_free\_days = previous\_consecutive\_free\_days_p \\
\quad cons\_days\_off = previous\_cons\_days\_off_p \\
\quad penalty_{p,C7} = 0 \\
\quad \forall d, (1 \leq d \leq D): \\
\quad \begin{cases}
x = |\{t \ \xi \ 1 \leq t \leq S \wedge schedule_{p,(d-1)*S+t} \neq 0\}| & \\
consecutive\_days + 1, & IF \quad (x \geq 1) \\
cons\_days\_off + 1, & IF \quad (x = 0 \wedge days\_off_{p,d} = 1) \\
\\
y = min\_cons\_free\_days_w - (cons\_free\_days + cons\_days\_off) & \\
penalty_{p,C7} + y, \ IF \quad (y > 0) & \\
cons\_free\_days = 0 & \\
cons\_days\_off = 0 & \\
IF \quad (x = 0 \wedge days\_off_{p,d} = 0) &
\end{cases}
\end{array}
\tag{9}
$$

$$
\begin{array}{l}
\forall p, (1 \leq p \leq P): \\
\quad w = w_p \\
\quad penalty_{p,C_8} = 0 \\
\quad x = max\_hours_w - AH_p \\
\quad y = previous\_hours_p + \sum_{t=1}^{T} shift\_duration_{s_t} * (1 - \delta_{schedule_{p,t},0}) \\
\quad penalty_{p,C_8} = y - x \qquad IF \quad (y - x > threshold\_hours)
\end{array}
\tag{10}
$$

Constraint 13 *Number of consecutive shift types*

For each shift type *shift*, a series of allowed sequences can be defined. In $consecutive\_shift_{w,shift}[i]$, i can take values from 1 to 10 and if $consecutive\_shift_{w,shift}[i] = 1$ the sequence i is allowed. The model, for example, supplies the possibility of defining 2, 4, and 6 as allowed sequences when $consecutive\_shift_{w,shift}[2] = consecutive\_shift_{w,shift}[4] = consecutive\_shift_{w,shift}[6] = 1$, and for all the other sequences the value is 0. Sequences of consecutive shifts at the end of the previous planning period can influence this constraint. They are denoted by $previous\_consecutive\_shifts_{p,s}$. The occurrence of an absence or illness day relaxes this constraint. When the result of adding the absence day(s) to a sequence satisfies the constraint, no penalty will be charged. Also, when the addition of absence days is not necessary to satisfy the constraint, the absences are ignored. It resembles the evaluation of Constraint 4, 5, 6, and 7 and is formally presented in (15).

Constraint 14 *Assign 2 free days after night shifts*

$$
\begin{array}{l}
\forall p, (1 \leq p \leq P): \\
\qquad \begin{array}{l}
w = w_p \\
penalty_{p,C_9} = 0 \\
x = min\_hours_w - AH_p \\
y = previous\_hours_p + \\
\sum_{t=1}^{T} shift\_duration_{s_t} * (1 - \delta_{schedule_{p,t},0}) \\
penalty_{p,C_9} = x - y \qquad\qquad\qquad\qquad IF \;\; (x - y > threshold\_hours)
\end{array}
\end{array}
\tag{11}
$$

$$
\begin{array}{l}
\forall p, (1 \leq p \leq P): \\
\qquad \begin{array}{l}
w = w_p \\
penalty_{p,C_{10}} = 0 \\
\\
\forall day, (1 \leq day \leq 7): \\
\\
\begin{cases}
x_{day} = max\_day_{w,day} \\
y_{day} = |\{(week, t) \; \xi \; 1 \leq week \leq D/7 \wedge 1 \leq t \leq S \\
\qquad\qquad \wedge schedule_{p,(week-1)*7*S+(day-1)*S+t} \neq 0\}| \\
\\
penalty_{p,C_{10}} + y_{day} - x_{day} \qquad\qquad\qquad IF \;\; (y_{day} > x_{day})
\end{cases}
\end{array}
\end{array}
\tag{12}
$$

Night shifts are all the shift types which begin before and end after 00:00. The set *night* contains the serial numbers of the corresponding shift types. When this constraint is valid ($night\_free_w = 1$), a night shift must be followed by another night shift or by two consecutive free days. An absence day counts for a free day. The constraint depends on daily sequences and the values will thus be transferred at the border of different planning periods. The number of free days after a night shift at the end of the previous planning period is given by $previous\_free\_after\_night_p$. The constraint is presented in (16).

Constraint 15 *Assign complete weekends*

Setting this constraint does not allow a shift on Saturday without one on the next Sunday or vice versa. It is denoted by $complete\_weekends_w$ for the entire work regulation. There is a possibility for redefining weekends, by either considering Friday and/or Monday as part of the weekend. The weekend definition is given by $weekend_w$, for which value 0 means a Saturday-Sunday weekend; 1 denotes a Friday-till-Sunday weekend; 2 stands for a Friday-till-Monday weekend, and 3 is a Saturday-till-Monday weekend. The complete weekend constraint will impose a shift to be planned on all the other weekend days as soon as there is an assignment on Saturday or Sunday. However, a scheduled shift on Friday or Monday does not require assignments on Saturday and Sunday. Again, absence days will be considered as free days or as working days whenever they relax the

$$
\begin{array}{l}
\forall p, (1 \leq p \leq P): \\
\quad w = w_p \\
\quad penalty_{p,C_{11}} = 0 \\
\quad \forall shift, (1 \leq shift \leq S): \\
\quad \begin{cases} x_{shift} = max\_shift_{w,shift} \\ y_{shift} = |\{d \ \xi \ 1 \leq d \leq D \wedge schedule_{p,(d-1)*S+t_{shift}} \neq 0\}| \\ \\ penalty_{p,C_{11}} + y_{shift} - x_{shift} \qquad\qquad IF \ \ (y_{shift} > x_{shift}) \end{cases}
\end{array} \tag{13}
$$

$$
\begin{array}{l}
\forall p, (1 \leq p \leq P): \\
\quad w = w_p \\
\quad penalty_{p,C_{12}} = 0 \\
\quad \forall week, (1 \leq week \leq D/7), \forall shift, (1 \leq shift \leq S): \\
\quad \begin{cases} x_{shift,week} = max\_shift_{w,shift,week} \\ y_{shift,week} = |\{d \ \xi \ 1 \leq d \leq 7 \wedge \\ \qquad schedule_{p,(week-1)*7*S+(d-1)*S+t_{shift}} \neq 0\}| \\ \\ penalty_{p,C_{12}} + y_{shift,week} - x_{shift,week}, IF \ (y_{shift,week} > x_{shift,week}) \end{cases}
\end{array} \tag{14}
$$

constraint most.
The schedule of the previous planning period can play a role in the evaluation of the constraint if the switch between planning periods happens in a weekend. In ANROM, it will only occur when the value of $weekend_w$ equals 2 or 3. We denote the weekend days which are transferred from the previous planning period by $previous\_sat_p$ and $previous\_sun_p$. These values are equal to 1 if there is at least one assignment on the corresponding day; it is 0 if the day is empty.The full constraint can be seen in (17).

Constraint 16 *No night shift before a free weekend*

Shifts which end after midnight cannot be scheduled before a free weekend when this constraint is valid, i.e. when $night\_weekend_w$ equals 1. For ordinary Saturday-Sunday weekends, this constraint requires that night shifts are not scheduled on Fridays when the entire weekend is free. Other definitions of weekends (e.g. Friday-Saturday-Sunday) restrict the schedule similarly. The constraint is illustrated in (18).

Constraint 17 *Assign identical shift types during the weekend*

$$
\begin{aligned}
&\forall p, (1 \leq p \leq P): \\
&\quad w = w_p \\
&\quad penalty_{p,C5} = 0 \\
&\quad \forall shift, (1 \leq shift \leq S): \\
&\quad \begin{cases}
cons\_days = previous\_consecutive\_shifts_{p,shift} \\
cons\_days\_off = previous\_consecutive\_days\_off_p \\
x = |\{d\ \xi\ 1 \leq d \leq D \wedge schedule_{p,(d-1)*S+t_{shift}} \neq 0\}| \\
cons\_days + 1 \qquad\qquad IF\ (x \neq 0) \\
cons\_days\_off + 1 \qquad\qquad IF\ (x = 0 \wedge days\_off_{p,d} = 1) \\
\begin{cases}
y = consecutive\_shift_{w,shift}[cons\_days] \\
z_i = consecutive\_shift_{w,shift}[cons\_days + i] \\
\qquad\qquad and\ 0 \leq i \leq cons\_days\_off \\
penalty_{p,C13} + 1,\ IF\ (y = 0 \wedge \sum_i z_i = 0) \\
consecutive\_days = 0 \\
cons\_days\_off = 0
\end{cases} \\
\qquad\qquad IF\ (x = 0 \wedge days\_off_{p,d} = 0)
\end{cases}
\end{aligned}
\tag{15}
$$

If this constraint (denoted by $identical\_weekend_w = 1$) is active, it creates a penalty when the shift types during the weekend days are not the same. No matter what the weekend definition is, whether it includes Friday and/or Monday, this constraint only looks at the shifts which are assigned on Saturday and Sunday. An absence will take a dummy value for this constraint, in order to generate the lowest possible penalty. The constraint is presented in (19).

Constraint 18 *Maximum number of consecutive working weekends*

This constraint limits the number of weekends in which duties are assigned with $max\_consecutive\_weekends_w$. It does not matter if the weekends are not completely scheduled. Only Saturdays and Sundays contribute to this constraint, even if Friday and/or Monday are considered part of the weekend. As is explained for other constraints on the order in which assignments may or may not be made (consecutiveness constraints), values are transferred to the next planning period. The number of consecutive weekends in person p's schedule at the end of the previous planning period is $previous\_consecutive\_weekends_p$. A formal representation can be seen in (20).

$$
\begin{array}{l}
\forall p, (1 \leq p \leq P): \\
\quad w = w_p \\
\quad penalty_{p,C14} = 0 \\
\quad cons\_days = previous\_free\_after\_night_p \\
\\
\quad IF\ (night\_free_w = 1) \\
\\
\quad \forall d, (1 \leq d \leq D): \\
\\
\quad \left\{ \begin{array}{l}
n = |\{t\ \xi\ 1 \leq t \leq S \wedge schedule_{p,(d-1)*S+t} \neq 0 \\
\qquad \wedge s_t \in night\}| \\
m = |\{t\ \xi\ 1 \leq t \leq S \wedge schedule_{p,(d-1)*S+t} \neq 0 \\
\qquad \wedge s_t \notin night\}| \\
\\
IF\ (last\_shift \in night) \\
\quad \left\{ \begin{array}{ll}
cons\_days + 1 & IF\ (n = 0 \wedge m = 0) \\
cons\_days = 0 & IF\ (n \neq 0) \\
penalty_{p,C14} + (2 - cons\_days) & \\
\qquad IF\ (n + m \neq 0 \wedge 2 - cons\_days > 0) &
\end{array} \right. \\
\\
ELSE\ IF\ (n \neq 0 \vee m \neq 0) \\
\quad \left\{ \begin{array}{l}
last\_shift = s_t \\
cons\_days = 0
\end{array} \right.
\end{array} \right.
\end{array}
\tag{16}
$$

$$
\begin{array}{l}
\forall p, (1 \le p \le P): \\
\quad w = w_p \\
\quad penalty_{p,C15} = 0 \\
\quad weekend\_days = previous\_weekend\_days \\
\\
\quad IF\ (complete\_weekends_w \neq 0):\ \forall wk, (1 \le wk \le D/7): \\
\\
\quad \left\{ \begin{array}{l}
absence\_fri = absence\_sat = absence\_sun = absence\_mon = 0 \\
fri = sat = sun = mon = 0 \\
fri = 1 \quad IF\ |\{t\ \xi\ 1 \le t \le S \wedge schedule_{p,((wk-1)*7+4)*S+t} \neq 0\}| \neq 0 \\
sat = 1 \quad IF\ |\{t\ \xi\ 1 \le t \le S \wedge schedule_{p,((wk-1)*7+5)*S+t} \neq 0\}| \neq 0 \\
sun = 1 \quad IF\ |\{t\ \xi\ 1 \le t \le S \wedge schedule_{p,((wk-1)*7+6)*S+t} \neq 0\}| \neq 0 \\
mon = 1\ IF\ |\{t\ \xi\ 1 \le t \le S \wedge schedule_{p,(wk*7*S+t} \neq 0\}| \neq 0 \\
\quad \left\{ \begin{array}{l}
fri = 1 \\
sat = previous\_sat_p \quad IF\ (wk = 1) \\
sun = previous\_sun_p
\end{array} \right. \\
mon = 1 \qquad\qquad IF\ (wk = D/7) \\
absence\_fri = days\_off_{p,(wk-1)*7+4}\ IF\ (fri = 0) \\
absence\_sat = days\_off_{p,(wk-1)*7+5}\ IF\ (sat = 0) \\
absence\_sun = days\_off_{p,(wk-1)*7+6}\ IF\ (sun = 0) \\
absence\_mon = days\_off_{p,wk*7}\ IF\ (mon = 0) \\
absence = absence\_sat + absence\_sun \\
\\
IF\ (weekend_w = 0): \\
penalty_{p,C15} + 1,\ IF\ (sat \neq sun \wedge absence = 0) \\
\\
IF\ (weekend_w = 1): \\
penalty_{p,C15} + 1,\ IF\ ((sat \neq sun \wedge absence = 0) \\
\quad \vee ((fri = 0 \wedge absence\_fri \neq 0) \wedge sat + sun = 2)) \\
\\
IF\ (weekend_w = 2): \\
penalty_{p,C15} + 1, \\
\quad IF\ ((sat \neq sun \wedge absence = 0) \vee (((fri = 0 \wedge absence\_fri \neq 0) \\
\quad \vee (mon = 0 \wedge absence\_mon \neq 0)) \wedge sat + sun = 2)) \\
\\
IF\ (weekend_w = 3): \\
penalty_{p,C15} + 1,\ IF\ ((sat \neq sun \wedge absence = 0) \\
\quad \vee ((mon = 0 \wedge absence\_mon \neq 0) \wedge sat + sun = 2))
\end{array} \right.
\end{array}
\tag{17}
$$

$$
\begin{array}{l}
\forall p, (1 \le p \le P): \\
\quad w = w_p \\
\quad penalty_{p,C16} = 0 \\
\quad IF\ (night\_weekend_w \neq 0):\ \forall week, (1 \le week \le D/7): \\
\quad \left\{ \begin{array}{l}
\begin{array}{rl}
fri & = |\{t\ \xi\ 1 \le t \le S \wedge schedule_{p,((week-1)*7+4)*S+t} \neq 0\}| \\
sat & = |\{t\ \xi\ 1 \le t \le S \wedge schedule_{p,((week-1)*7+5)*S+t} \neq 0\}| \\
sun & = |\{t\ \xi\ 1 \le t \le S \wedge schedule_{p,((week-1)*7+6)*S+t} \neq 0\}|
\end{array} \\
IF\ (weekend_w = 0 \vee weekend_w = 3) \\
\quad \left\{ \begin{array}{l}
n = |\{t\ \xi\ 1 \le t \le S \wedge schedule_{p,((week-1)*7+4)*S+t} \neq 0 \\
\qquad\qquad \wedge\ s_t \in night\}| \\
penalty_{p,C16} + 1, \qquad IF\ (sat + sun = 0 \wedge n = 1)
\end{array} \right. \\
IF\ (weekend_w = 1 \vee weekend_w = 2) \\
\quad \left\{ \begin{array}{l}
n = |\{t\ \xi\ 1 \le t \le S \wedge schedule_{p,((week-1)*7+3)*S+t} \neq 0 \\
\qquad\qquad \wedge\ s_t \in night\}| \\
penalty_{p,C16} + 1, \qquad IF\ (fri + sat + sun = 0 \wedge n = 1)
\end{array} \right.
\end{array} \right.
\end{array}
\tag{18}
$$

$$
\begin{array}{l}
\forall p, (1 \le p \le P): \\
\quad w = w_p \\
\quad penalty_{p,C17} = 0 \\
\quad IF\ (identical\_weekend_w = 1):\ \forall week, (1 \le week \le D/7): \\
\quad \left\{ \begin{array}{l}
\forall t, (1 \le t \le S): \\
\quad \left\{ \begin{array}{ll}
sat = t & IF\ (schedule_{p,((week-1)*7+5)*S+t} \neq 0) \\
sun = t & IF\ (schedule_{p,((week-1)*7+6)*S+t} \neq 0) \\
absence\_sat = days\_off_{p,(week-1)*7+5} & IF\ (sat = 0) \\
absence\_sun = days\_off_{p,(week-1)*7+6} & IF\ (sun = 0)
\end{array} \right. \\
penalty_{p,C17} + 1,\ IF\ (sat \neq sun \wedge absence\_sat + absence\_sun = 0)
\end{array} \right.
\end{array}
\tag{19}
$$

$\forall p, (1 \leq p \leq P)$:

$w = w_p$
$consecutive\_weekends = previous\_consecutive\_days_p$
$penalty_{p,C18} = 0$

$\forall week, (1 \leq week \leq D/7)$:

$$\begin{cases} sat = |\{t\ \xi\ 1 \leq t \leq S \wedge schedule_{p,((week-1)*7+5)*S+t} \neq 0\}| \\ sun = |\{t\ \xi\ 1 \leq t \leq S \wedge schedule_{p,((week-1)*7+6)*S+t} \neq 0\}| \\ consecutive\_weekends + 1, \qquad IF\ (sat + sun > 0) \\ \left.\begin{array}{l} x = consecutive\_weekends - max\_consecutive\_weekends_w \\ penalty_{p,C18} + x, \qquad IF\ (x > 0) \\ consecutive\_weekends = 0 \end{array}\right\} \\ \qquad\qquad\qquad\qquad\qquad\qquad IF\ (sat + sun = 0) \end{cases} \tag{20}$$

Constraint 19 *Maximum number of working weekends in 4 weeks*

The constraint is a restriction on weekend work during periods of 4 consecutive weeks, provided that at least one of the 4 weeks belongs to the planning period. The maximum number of weekends is given by $max\_weekends\_4\_weeks_w$. Suppose, for example, that we have a planning period of x weeks. The constraint will be evaluated in x overlapping periods, from the period which starts 3 weeks before the current one, up until the period which ends with the last week of the current period. The number of working weekends for person p in the previous planning period is given by $previous\_weekends\_3_p$, $previous\_weekends\_2_p$, and $previous\_weekends\_1_p$. It is a very specific request which was implemented to satisfy the needs of particular users of the software based on an earlier version of ANROM and is formally illustrated in (21).

$$
\begin{aligned}
&\forall p, (1 \leq p \leq P): \\
&\quad w = w_p \\
&\quad penalty_{p,C19} = 0 \\
&\quad \forall period, (1 \leq period \leq D/7): \\
&\quad \begin{cases}
n = 4 - period \\
weekends = \begin{cases} previous\_weekends\_n_p \ IF \ (n > 0) \\ 0 \qquad\qquad\qquad\qquad ELSE \end{cases} \\
\forall week, (-n \leq week \leq -n + 4 \wedge week > 0): \\
\quad \begin{cases}
sat = |\{t \ \xi \ 1 \leq t \leq T \wedge schedule_{p,((week-1)*7+5)*S+t} \neq 0\}| \\
sun = |\{t \ \xi \ 1 \leq t \leq T \wedge schedule_{p,((week-1)*7+6)*S+t} \neq 0\}| \\
weekends + 1, \qquad\qquad IF \ (sat + sun > 0)
\end{cases} \\
x = weekends - max\_weekends\_4\_weeks_w \\
penalty_{p,C19} + x, \qquad IF \ (x > 0)
\end{cases}
\end{aligned}
\tag{21}
$$

Constraint 20 *Maximum number of assignments on bank holidays*

Unlike most of the other constraints this constraint is cumulative. Bank holidays are recorded over a longer period than the planning period only. For each person, the number of cumulative assignments on bank holidays is denoted by $previous\_bank_p$. The maximum number of assignments on bank holidays per work regulation is given by $bank\_holidays_w$. A structure *bank* is an array of length D and it has value 1 for bank holidays and value 0 for other days. Usually hospitals prefer to limit the number of assignments on bank holidays during an entire year with this value. The constraint is formally demonstrated in (22).

$$
\begin{aligned}
&\forall p, (1 \leq p \leq P) : \\
&\qquad w = w_p \\
&\qquad bank\_days = previous\_bank_p \\
&\qquad penalty_{p,C20} = 0 \\
&\qquad \forall d, (1 \leq d \leq D \wedge bank[d] = 1) : \\
&\qquad \begin{cases} x = |\{t \ \xi \ 1 \leq t \leq S \wedge schedule_{p,(d-1)*S+t} \neq 0\}| \\ bank\_days + 1 \qquad\qquad\qquad\qquad IF \ (x > 0) \end{cases} \\
&\qquad y = bank\_days - bank\_holidays_w \\
&\qquad penalty_{p,C20} = y \qquad\qquad IF \ (y > 0)
\end{aligned}
\tag{22}
$$

Constraint 21 *Restriction on the succession of shift types*

The constraint on the minimum time between shift types already restricts some sequences of constraints. However, the current constraint can explicitly forbid particular combinations of shift types. Unlike the constraint on minimum time between assignments, this constraint evaluates shifts which are scheduled on consecutive days. A scheduled shift is connected to the day at which the shift starts. The succession constraint also provides the possibility of forbidding certain shifts after a free day or even free days after certain shifts. The restrictions are denoted by $succession_w$, a two dimensional structure with a column and row for each shift type in addition to one for an empty day. The elements in $succession_w$ are 0 when the column shift cannot be scheduled after the row shift, and 1 when there is no restriction on the succession. Only the last day of the previous planning period can influence the evaluation of the constraint in the current period. The parameter $last\_day_{p,t}$, has value 1 when the corresponding assignment unit t on the last day of person p's previous planning period is occupied. A real-world example demonstrating this constraint is given in Table

3. The combinations of letters in the rows and columns (SE, EE, etc) are abbreviations of shift types in a practical hospital application. The corresponding shift types have been presented in Table 2.

Succession	-	SE	EE	SD	E	D	SL	L	LL	N
-	v	v	v	v	v	v	v	v	v	v
SE	v	v	v	v	v	v	v	v	v	v
EE	v	v	v	v	v	v	v	v	v	v
SD	v			v	v	v	v	v	v	v
E	v			v	v	v	v	v	v	v
D	v					v	v	v	v	v
SL	v						v	v	v	v
L	v						v	v	v	v
LL	v								v	v
N	v									v

Table 3. Allowed successions of shift types on consecutive days are represented by 'v', '-' denotes a day on which nothing is scheduled

Personal constraints It is often possible for individual personnel members to make agreements with the personnel manager or head nurse. External or private obligations do not fall under the category of hard constraints. They can theoretically be cancelled in emergency situations. However, there are several possibilities of giving extra weight to a personal obligation. The reason for the absence can be taken into account in addition to the importance of the external commitment. Such situations are represented by the following constraints.

Constraint 22 *Day off*

Anything that prevents the personnel member from being at work can be handled as a day off in the cost function. Depending on the reason for the day off, some types of requests for absence will affect the value which is set for some of the other constraints (see Constraint 3, 7, 8, 9, etc). Every person has the right to take holiday during the working year. We do not want to generate penalties for undertime when a person takes holidays in the same period.
Illness, refresher courses, compensation, and occasional family reasons are all examples of day off types which can be placed in the schedule. The requested days off for person p at day d are denoted by 1 in $days\_off_{p,d}$. The program foresees a possibility of using an extra weight (to multiply the weight factor with) for imperative needs. When the extra weight is valid, we find a 1 in the structure $extra_{p,d}$. The value of the extra weight is the same for the entire ward: $extra\_penalty$. The system will thus distinguish between strong and weak day off requests and penalise correspondingly. This constraint is formally defined in (24).

$$
\begin{aligned}
&\forall p, (1 \le p \le P) : \\
&\qquad w = w_p \\
&\qquad succession = succession_w \\
&\qquad last\_day = last\_day_p \\
&\\
&\qquad penalty_{p,C21} \\
&\qquad = |\{(t,u)\ \xi\ (1 \le t \le S \wedge 1 \le u \le t)\ \wedge \\
&\qquad\quad (last\_day_t \ne 0 \wedge schedule_{p,u} \ne 0) \wedge succession_{t,u} = 0\}| \\
&\qquad + |\{(t,u)\ \xi\ (1 \le t \le T - S \wedge t < u \le t + S)\ \wedge \\
&\qquad\quad (schedule_{p,t} \ne 0 \wedge schedule_{p,u} \ne 0) \wedge succession_{s_t,s_u} = 0\}| \\
&\qquad + |\{t\ \xi\ (1 \le t \le S) \wedge succession_{t,-} = 0\ \wedge \\
&\qquad\quad last\_day_t \ne 0 \wedge |\{u\ \xi\ 1 \le u \le S \wedge schedule_{p,u} \ne 0\}| = 0\}| \\
&\qquad + |\{t\ \xi\ (1 \le t \le T - S) \wedge succession_{s_t,-} = 0 \wedge schedule_{p,t} \ne 0 \\
&\qquad\quad \wedge |\{u\ \xi\ 1 \le u \le S \wedge schedule_{p,(t/S+1)*S+u} \ne 0\}| = 0\}| \\
&\qquad + |\{t\ \xi\ (1 \le t \le S) \wedge succession_{-,t} = 0\ \wedge \\
&\qquad\quad schedule_{p,t} \ne 0 \wedge |\{u\ \xi\ 1 \le u \le S \wedge last\_day_u \ne 0\}| = 0\}| \\
&\qquad + |\{t\ \xi\ (S \le t \le T) \wedge succession_{-,s_t} = 0 \wedge schedule_{p,t} \ne 0 \\
&\qquad\quad \wedge |\{u\ \xi\ 1 \le u \le S \wedge last\_day_{(t/S-1)*S+u} \ne 0\}| = 0\}|
\end{aligned}
\tag{23}
$$

$$
\begin{aligned}
&\forall p, (1 \le p \le P) : \\
&\qquad penalty_{p,C22} = 0 \\
&\qquad \forall d, (1 \le d \le D) : \\
&\qquad \begin{cases}
x = |\{t\ \xi\ 1 \le t \le S \wedge schedule_{p,(d-1)*S+t} \ne 0\}| \\
IF\ (days\_off_{p,d} = 1 \wedge x > 0) \\
\qquad penalty_{p,C22} + \begin{cases} 1 & IF\ (extra_{p,d} = 0) \\ extra\_penalty & ELSE \end{cases}
\end{cases}
\end{aligned}
\tag{24}
$$

Constraint 23 *Shifts off*

People can avoid certain shifts on a particular day of the planning period and then $shift\_off_{p,t}$ equals 1. For the rest of the assignment units of the planning period, the value is 0. It is recommended to avoid conflicts with certain activities in the personal agenda by blocking small parts of the planning period. The idea is the same as in patterns, but this constraint is not cyclic. Also, the feature to attach a stronger weight to some requests (as explained with respect to Constraint 22), exists for this constraint. Those requests which require a stronger penalty have 1 in $extra\_shift_{p,t}$ The formal definition can be seen in (25).

$$
\begin{aligned}
&\forall p, (1 \leq p \leq P): \\
&\quad penalty_{p,C23} = 0 \\
&\quad \forall t, (1 \leq t \leq T): \\
&\quad \begin{cases} IF\ (shift\_off_{p,t} = 1 \wedge schedule_{p,t} \neq 0) \\ \quad penalty_{p,C23} + \begin{cases} 1 & IF\ (extra\_shift_{p,t} = 0) \\ extra\_penalty & ELSE \end{cases} \end{cases}
\end{aligned} \tag{25}
$$

Constraint 24 *Requested assignments*

There are cases in which a person wants to be assigned to a specific shift type on a certain day. The set of required assignments (corresponding to assignment unit t) for person p is denoted by 1 in $requested\_assignment_{p,t}$. For the other assignment units, the value is 0. As explained with respect to Constraint 23, there is a possibility for giving a higher or lower importance to each requested assignment. For the required assignments with a higher importance, the structure $extra\_requested\_shift_{p,t}$ has the value 1. The system applies the same weight factor and multiplication factor for violations on personal constraints as explained in the day off constraint (Constraint 22). A formal illustration is presented in (26).

$$
\begin{aligned}
&\forall p, (1 \leq p \leq P): \\
&\quad penalty_{p,C24} = 0 \\
&\quad \forall t, (1 \leq t \leq T): \\
&\quad \begin{cases} IF\ (requested\_assignment_{p,t} = 1 \wedge schedule_{p,t} \neq 0) \\ \quad penalty_{p,C24} + \begin{cases} 1 & IF\ (extra\_requested\_shift_{p,t} = 0) \\ extra\_penalty & ELSE \end{cases} \end{cases}
\end{aligned} \tag{26}
$$

Constraint 25 *Tutorship*

There exists a possibility of defining a tutor for a personnel member who cannot work alone: $tutor_p$. This constraint implies that the tutor has to be working whenever the other person is. The same concept can be used for any set of people who want to work at the same time (e.g. tutees, car-poolers, etc). ANROM does not generate a penalty when the tutor is working during a free moment of the tutee, and neither when the tutor's shift overlaps completely with the tutee's shift. When two people are required at the same time all the time, they can be set as each other's tutor. The constraint is presented in (27).

$$
\begin{aligned}
&\forall p, (1 \leq p \leq P) : \\
&\qquad tutor = tutor_p \\
&\qquad penalty_{p,C25} = |\{t \ \xi \ 1 \leq t \leq T \wedge schedule_{p,t} \neq 0 \wedge cover_{tutor,t} = 0\}| \\
&\qquad with \\
&\qquad cover_{tutor,t} = |\{u \ \xi \ (t/S) * S \leq u \leq (t/S + 1) * S \\
&\qquad\qquad \wedge \ shift\_start_{s_u} \leq shift\_start_{s_t} \wedge shift\_end_{s_u} \geq shift\_end_{s_t}\}|
\end{aligned}
\tag{27}
$$

Constraint 26 *People not allowed to work together*

This constraint applies the same idea as the above. It only prevents the two people involved from being present in the ward at the same time. The person $not\_together_p$ should not work when p is at work. The constraint is often used in order to provide a maximal availability of people with equal skills. Other applications are those in which family members prefer to alternate their working time in order to take care of the children. All the assignments which are overlapping in time in their schedules violate this constraint which is formally presented in (28).

$$
\begin{aligned}
&\forall p, (1 \leq p \leq P) : \\
&\qquad not = not\_together_p \\
&\qquad penalty_{p,C26} = |\{t \ \xi \ 1 \leq t \leq T \wedge schedule_{p,t} \neq 0 \wedge overlap_{not,t} \neq 0\}| \\
&\qquad with \\
&\qquad overlap_{not,t} = |\{u \ \xi \ (t/S) * S \leq u \leq (t/S) * S + S \\
&\qquad\qquad\qquad \wedge (shift\_start_{s_t} \leq shift\_start_{s_u} < shift\_end_{s_t} \\
&\qquad\qquad\qquad \vee \ shift\_start_{s_t} < shift\_end_{s_u} \leq shift\_end_{s_t})\}|
\end{aligned}
\tag{28}
$$

References

1. Greet Vanden Berghe. *An Advanced Model and Novel Meta-Heuristic Solution Methods to Personnel Scheduling in Healthcare.* PhD thesis, University of Gent, Belgium, 2002.