Template and constraint-based models for the optimization of schedules, applied to Netherlands Railways (NS)' crew planning

Veerle E. van den Hurk[1], Egon L. van den Broek[1][0000−0002−2017−0141], Marjan van den Akker[1][0000−0002−7114−0655], and Erwin Abbink[2]

[1] Department of Information and Computing Sciences, Utrecht University, Utrecht, the Netherlands
{v.e.vandenhurk,j.m.vandenakker}@uu.nl and vandenbroek@acm.org
[2] Netherlands Railways (NS), Utrecht, the Netherlands
erwin.abbink@ns.nl

Abstract. Planning and scheduling remains among Artificial Intelligence's biggest challenges. Especially, in real-world scenarios, when multiple types of constraints need to be met: available resources, labor law and regulations, and people's preferences. To accommodate these constraints and generate tractable, optimal schedules, we propose an Integer Linear Programming model (ILP) that handles rostering as an assignment problem. Instead of duties, this uses duty templates to generate cyclic rosters, i) unveiling redundancy in the set of constraints, ii) realizing speed improvement by removing symmetries in the solution space, and iii) allowing an integrated approach to rostering. The question posed by Netherlands Railways (NS) whether it is possible to create feasible rosters using those duty templates is answered positively by solving the ILP's feasibility problem. However, an NS roster expert considered the solutions unattractive. Therefore, we introduce soft constraints, moving from a feasibility to an optimization problem. The updated model delivered explainable, tractable, optimally balanced workload, and more attractive schedules in a complex, real-world scenario.

Keywords: Planning and Scheduling · Optimization · Constraint-Based Modelling · Real-world · Personnel

1 Introduction

Starting with their birth, the history of Artificial Intelligence (AI) and Operations Research (OR) has always been intertwined [13, 18]. However, throughout the years their relation has ranged from close to distant [15, 23]. On the one hand, AI and OR share some of their main challenges [15], such as i) planning and scheduling, ii) search iii) heuristics, iv) constraint-based reasoning, and v) machine learning. Moreover, both AI and OR study complex problems in various domains, such as healthcare, transport, and energy. On the other hand, AI and OR have essential differences [15]. OR delivers tractable models (e.g., using

Mixed Integer Program (MIP)). For well-defined problem spaces, it can deliver optimal solutions. However, the models are rigid, often lacking expressive power. In contrast, AI embraces a variety of rich knowledge representations, which can function at different levels, ranging from symbolic to semantic (e.g., deep learning). However, AI 's can deliver intractable, black box models [20]. Interestingly, AI and OR have to potential counter each others weaknesses [13,15,23]. In other words, it seems feasible that AI helps OR and OR helps AI . Hence, a joint integral, overarching framework remains worth exploring [12, 14]. This is especially the case for real world problems, such as planning and scheduling, which require the best of both worlds: models need to be both expressive and explainable [20], while delivering (near) optimal solutions. Par excellence and in contrast with regular planning challenges, this is the case with personnel (or: crew) scheduling, as this directly impacts people's experience.

On the one hand, AI helps OR : Bengio et al. [5] survey the recent attempts at leveraging machine learning to solve combinatorial optimization problems. The authors advocate for pushing further the integration of machine learning and combinatorial optimization. In more recent years, AI techniques are used to further improve the MIP solvers' branch-and-bound algorithm, as is surveyed recently by Scavuzzo et al. [21]. On the other hand, OR helps AI : In 1999, Vossen et al. [25] already explored the use of Integer Linear Program (ILP) to solve AI planning problems. Tsay et al. [24] introduce a class of mixed-integer formulations for trained ReLU neural networks and use it to find adversarial examples. Recently, Aghaei et al. [4] proposed mixed-integer optimization-based techniques for learning optimal binary classification trees. So, AI and OR 's combined strengths shows a promising area of research in AI 's current era, where model performance alone is not considered sufficient. Also aspects such as user experience [7] and fairness [7, 19] need to be considered next to tractability, explainability [22], and computational complexity [8].

In this article, we study one of AI and OR 's main challenges: planning and scheduling, which is widely studied across several domains. One of the more prominent application areas of planning and scheduling is the scheduling of personnel. Due to the increased importance of employee preferences and satisfaction, this challenge's complexity has significantly grown over the years. Van den Bergh et al. [6] described that the interest in personnel scheduling problems can be attributed to economic considerations, as reducing labor costs is beneficial for many companies. Additionally, including personnel preferences in the crew scheduling problem not only optimizes services, it also leads to a higher job satisfaction and a lower level of employee sickness. Hanne et al. [16] described this as "happy staff means happy customers".

To give insight in real-world crew scheduling's complexity, we consider the crew planning process at Netherlands Railways (NS). Here, crew planning is currently split up into two phases:

1. *scheduling*: The total set of duties (i.e., sequences of tasks corresponding to a day of work for a single crew member) for a whole year are created for each crew base simultaneously [2].

2. *rostering*: At each crew base, annual rosters are created from the assigned set of duties. A roster specifies, for each crew member and day, whether the crew member works on that day, and if so, which duty is to be performed [19].

To arrive at a fair division of the preferred (*sweet*) and less preferred (*sour*) work among the different crew bases, so-called *sharing sweet and sour*-rules (in Dutch: lusten en lasten delen) are taken into account in the scheduling phase [3]. Decision Support Systems have been developed to at least partially automate the rostering process (e.g., see [7, 17]). However, up to this date these systems have not been successfully implemented in real-world practice. Hence, the rostering phase is still carried out *manually* and is split up in three parts.

Despite the *sharing sweet and sour*-rules, a fair assignment and realizing working times comparable to the annual roster remained challenging. Consequently, NS explores both alternative and complementary strategies. The most prominent one being a new crew planning process based on *duty templates*. Such a duty *template* is characterized by a day and time window in which an employee can be scheduled to work, and acts as a placeholder for a duty in a roster [19]. In the scheduling phase, these duty templates will be generated instead of duties themselves. In the rostering phase, per crew base, these templates together with days off must be formed into cyclic annual template-based rosters. A week before operation, the actual duties and time of work are published to the crew members, guaranteed to be within the template rosters. Because the duties are generated close to the day of operation, they can be determined more accurately and less rescheduling needs to be done afterwards. The generation of duties is based on the template rosters and, thus, can take into account crew members' qualifications and individual scores on the *sharing sweet and sour*-rules [19]. When considering the process based on duty templates, NS' main question is: *Using templates instead of duties, is it feasible to create rosters obeying all rules?*

Next, we elaborate on the NS rostering problem and the notation used. Our cyclic crew rostering model using templates is presented in Section 3. In Section 4 this model is validated on real-life instances the NS provided, providing us with an answer to the question posed above. We close this paper with Section 5, which provides a discussion, reflecting on the work presented. Additionally, an Appendix defining functions used in Section 3 is presented.

2 Problem description and notation

Using templates corresponding to a crew base at NS, an instance of the Cyclic Crew Rostering Problem, is given by:

- For each template $b \in B$:
  - start time S_b and end time E_b in minutes (not given modulo 1440)
  - $K_b \in \{0, 1\}$ denotes whether $E_b > 1560$
  - $M_b \in \{0, 1\}$ denotes whether $E_b > 1440$
  - $N_b \in \{0, 1\}$ denotes whether $b \in B$ is a *night template* (fits a night duty)

  - Furthermore, each $b \in B$ is part of exactly one of the subsets B_j (templates that need to be assigned to day j)
- Each crew member $i \in C$ is part of exactly one of the roster groups $C_1, C_2, ..., C_g$.
- For each roster group c, the number of S and WTV days per $|C_c|$ weeks are given by s_c and wtv_c.

With three types of rest days:

- R: normal rest days;
- WTV: reduction in working hours (in Dutch: *werktijdverkorting*); and
- S: compensation days for irregular work and part time work,

which together make the set $D = \{R, WTV, S\}$. We can represent a roster for a single crew base as a matrix, as is shown in Figure 1. As the roster is cyclic, members of a single roster group rotate through the same roster. Rosters have to comply to rules, specified as hard constraints, see Table 1. The set of templates B needs to be assigned to the cells of the matrix, including appropriate R, WTV, and S days, in a way that satisfies these rules. Satisfying rules when considering templates means that each duty that can possibly be the content of a template respects the rules. Note that less roster rules need to be considered compared to considering duties, as the templates have no content.

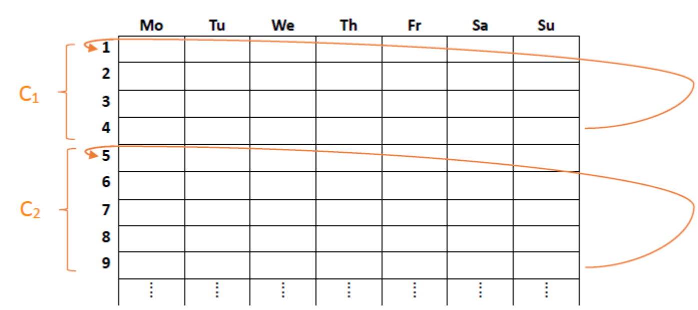

Fig. 1: Example of a roster for a single crew base. Rows $i \in C$ represent the crew members, columns $j \in W = \{1, 2, 3, 4, 5, 6, 7\}$ represent weekdays: Monday to Sunday. Lines corresponding to a single roster group are the subrosters, which are cyclic: the last cell of a subroster is succeeded by the first cell of the subroster.

We also consider 4 types of roster groups: *mix, early, late* and *elderly*. Each type obeys additional rules regarding working times (e.g., rosters of type *early* contain no duties ending after 18.00). These additional group-based constraints can be incorporated into the model presented in the next section. Furthermore, we assume no template starts before 04.00.

Table 1: Roster rules, according to the NS collective labour agreement [1]

Daily rest time	$\geq$ 12 hours (uninterrupted)
Weekly rest time	Every period of 14 $\times$ 24 hours needs to contain an uninterrupted rest period of 72 hours or 2 $\times$ 36 hours each.
Types of rest days	A WTV day covers a calendar day that starts not later than 02.00. Each calendar week is assigned two R days on average and needs to have at least one R day. R and S days contain $\geq$ 30 hours when succeeding a duty, otherwise $\geq$ 24 hours.
Consecutive duties	$\leq$ 7
Nigth duty	Rest time after a duty that ends after 02.00 needs to be $\geq$ 14 hours. After series of $\geq$ 3 night duties, a rest period of $\geq$ 46 hours is mandatory.
Red Weekend	A period of $\geq$ 60 hours, which covers Saturday 00:00 until Monday 04:00. Present at least one out of every three weekends.

3 Modelling

To answer the feasibility question posed by NS for cyclic crew rostering, we adopted an ILP , without objective at first. Par excellence, an ILP can generate tractable, rule-based models. Rules such as those in the collective NS labor agreement (see Table 1) can be formulated as hard constraints, allowing and ILP to provide exact solutions. In turn, this allows to answer our feasibility question. Next, we formulate the assignment problem, propose a naive model, and shrink its solution space.The proposed model assigns days off and templates to the roster of a crew base simultaneously. This integrates all three parts of the current roster procedure at NS, contrary to earlier work (e.g., [17], [7]).

3.1 Mathematical formulation

Table 2 provides the ILP model's binary variables and its helper functions, which are defined the Appendix. We can get the separate elements, either row or column, corresponding to $h(i,j)$ by writing $h_1(i,j)$ (row) or $h_2(i,j)$ (column). Further, we write $x_{h(i,j)} = x_{h_1(i,j)h_2(i,j)}$ and $h^1(i,j) = h(i,j)$.

The following constraint guarantees that a cell cannot be empty:

$$\sum_{b \in B_j} x_i^b + \sum_{d \in D} x_{ij}^d = 1 \qquad \forall (i,j) \in C \times W, \tag{1}$$

as it assigns either a template or a rest day.

Moreover, each template is assigned to exactly one row:

$$\sum_{i \in C} x_i^b = 1 \qquad \forall j \in W, \forall b \in B_j, \tag{2}$$

taking into account that each template is already assigned to a certain weekday.

Table 2: Variables and helper functions

x_i^b	Equals 1 if template b is assigned to row i, and 0 otherwise.
x_{ij}^d	Equals 1 if rest day d is assigned to cell (i, j), and 0 otherwise.
m_{ij}	Can be 1 if a rest period of at least 36 hours starts after the template on (i, j), if not it equals 0. If no template is assigned to (i, j), it also equals 0.
p_{ij}	Can be 1 if a rest period of at least 72 hours starts after the template on (i, j), if not it equals 0. If no template is assigned to (i, j), it also equals 0.
z_{ij}^a	Equals 1 if all cells $h(i, j)$ up to and including $h^a(i, j)$ are assigned a type from $\{R, S\}$ and surrounding cells (i, j) and $h^{a+1}(i, j)$ are not, and 0 otherwise.
v_{ij}	Can be 1 if a rest period of at least 46 hours starts after the template on $h^2(i, j)$, if not it equals 0.
RW_i	Can be 1 if row i contains a Red Weekend (RW), if not it equals 0.
$h^n(i_1, j_1)$	Returns cell (i_2, j_2) taking place n days after (or in the case of $n = -1$, the day before) (i_1, j_1) in the same subroster.
$t((i_1, j_1),(i_2, j_2))$	Returns the time in minutes between the end of the template on cell (i_1, j_1) end the beginning of the template on cell (i_2, j_2), or returns large enough (specified in Appendix) number if one of these cells is not assigned a template.
$t^U((i_1, j_1),(i_2, j_2))$	Modified version of $t((i_1, j_1),(i_2, j_2))$ where we only consider templates $b \in B$ on (i_1, j_1) which have a certain property indicated by the binary variable U_b, otherwise it returns a large enough number.

The number of WTV and S days per roster group is fixed by

$$\sum_{i \in C_c} \sum_{j \in W} x_{ij}^S = s_c \qquad \text{and} \qquad \sum_{i \in C_c} \sum_{j \in W} x_{ij}^{WTV} = wtv_c \qquad \forall c = 1, ..., g. \tag{3}$$

Consecutive templates have at least 720 minutes between them:

$$t((i, j), h(i, j)) \geq 720 \qquad \forall (i, j) \in C \times W. \tag{4}$$

Note that a rest period of ≥ 36 (72) hours corresponds to either 1 (2) type of rest day(s) with sufficient rest time between the templates around this rest day(s) or ≥ 2 (3) rest days. Then, if the considered periods do not correspond to rest of respectively 2160 or 4320 minutes at least, $m_{ij} = 0$ and $p_{ij} = 0$ are fixed by the following constraints $\forall (i, j) \in C \times W$:

$$\sum_{b \in B_j} x_i^b \geq m_{ij} \tag{5}$$

$$\sum_{d \in D} x_{h(i,j)}^d \geq m_{ij} \tag{6}$$

$$t((i, j), h^2(i, j)) \geq 2160 m_{ij} \tag{7}$$

$$\sum_{b \in B_j} x_i^b \geq p_{ij} \tag{8}$$

$$\sum_{d \in D} x_{h(i,j)}^d \geq p_{ij} \tag{9}$$

$$\sum_{d \in D} x_{h^2(i,j)}^d \geq p_{ij} \tag{10}$$

$$t((i,j), h^3(i,j)) \geq 4320 p_{ij}. \tag{11}$$

Note that $p_{ij} = 1$ implies $m_{ij} = 1$, as a rest period of 36 hours starting at (i,j) is also contained in the period of 72 hours. Then,

$$\sum_{n=-1}^{12} m_{h^n(i,j)} + \sum_{n=-1}^{11} p_{h^n(i,j)} \geq 2 \qquad \forall (i,j) \in C \times W, \tag{12}$$

secures that each period of 14 days contains either 2 rest periods of ≥ 36 hours or 1 of ≥ 72 hours. Note that this formulation allows the rest period to not be fully contained in the considered horizon: it allows a rest period to cross the boundaries of the considered period of 14 days, which loosens the rule stated.

With respect to rest days, we define the following constraints. If a template ends after 02.00, the following holds:

$$1 - \sum_{b \in B_j} x_i^b K_b \geq x_{h(i,j)}^{WTV} \qquad \forall (i,j) \in C \times W, \tag{13}$$

which forbids assigning a WTV day after. Next, to secure an average of two rest days per week, we define:

$$\sum_{i \in C_c} \sum_{j \in W} x_{ij}^R \geq 2|C_c| \qquad \forall c = 1, ..., g, \tag{14}$$

which defined each roster group c having at least $2|C_c|$ R days in their subroster. Last, the mandatory 1 rest day per week is modelled as:

$$\sum_{j \in W} x_{ij}^R \geq 1 \qquad \forall i \in C. \tag{15}$$

Further, let A be the set corresponding to possible number of consecutive R, S days and assume $A = \{1, 2, 3, 4, 5, 6, 7\}$. Then, we set $z_{ij}^a = 1$ in (16) if we have i) all cells $h(i,j)$ up to and including $h^a(i,j)$ are assigned a type from $\{R, S\}$ and ii) surrounding cells (i,j) and $h^{a+1}(i,j)$ are not assigned a type from $\{R, S\}$:

$$\begin{aligned} z_{ij}^a \geq \sum_{n=1}^{a} \sum_{d \in \{R,S\}} x_{h^n(i,j)}^d - a + 1 - \sum_{d \in \{R,S\}} x_{ij}^d \\ - \sum_{d \in \{R,S\}} x_{h^{a+1}(i,j)}^d \qquad \forall (i,j) \in C \times W, \forall a \in A. \end{aligned} \tag{16}$$

Then, the rules corresponding to norms of R, S are covered in:

$$t^{WTV}((i,j), h^{a+1}(i,j)) \geq z^a_{ij}\left(360 + 1440a\right) - 360x^{WTV}_{ij} \qquad \forall (i,j) \in C \times W, \forall a \in A, \tag{17}$$

where we check whether or not the surrounding cells are assigned templates or WTV days, which cover enough rest time only if the corresponding $z^a_{ij} = 1$. If a WTV day precedes an R, S day, this R, S needs to add 24 hours to the rest period, corrected by Equation (17)'s last term, starting at the end of WTV (00.00). If a WTV day succeeds an R, S day, the rest period corresponding to R, S ends at 02.00. This is taken into account in t^{WTV}, see Appendix.

To guarantee that each 8 day period contains ≥ 1 day of type D, we model:

$$\sum_{n=0}^{7} \sum_{d \in D} x^d_{h^n(i,j)} \geq 1 \qquad \forall (i,j) \in C \times W. \tag{18}$$

Moreover, when a template ends after 02.00, we define

$$t^K((i,j), h(i,j)) \geq 840 \qquad \forall (i,j) \in C \times W, \tag{19}$$

which guarantees ≥ 14 hours rest time. We make sure each series of ≥ 3 night templates is followed by ≥ 46 hours rest by

$$\sum_{n=0}^{2} \sum_{b \in B_{h_2^n(i,j)}} x^b_{h_1^n(i,j)} N_b \leq 2 + \sum_{d \in D} x^d_{h^3(i,j)} + \sum_{b \in B_{h_2^3(i,j)}} x^b_{h_1^3(i,j)} N_b \qquad \forall (i,j) \in C \times W, \tag{20}$$

which models that each series of three or more night templates is succeeded by either another night template or a type of rest day, and

$$\left(\sum_{n=0}^{2} \sum_{b \in B_{h_2^n(i,j)}} x^b_{h_1^n(i,j)} N_b\right) + \sum_{d \in D} x^d_{h^3(i,j)} \leq 3 + v_{ij} \qquad \forall (i,j) \in C \times W, \tag{21}$$

which models whenever a type of rest day is chosen in the previous case, the rest period needs to contain ≥ 46 hours together with

$$t(h^2(i,j), h^4(i,j)) \geq 2760 v_{ij} \qquad \forall (i,j) \in C \times W. \tag{22}$$

Note that adding $\sum_{d \in D} x^d_{h^3(i,j)} \geq v_{ij}$ is redundant as $v_{ij} = 0$ is then already set by (21). Further, it may happen that out of 4 consecutively assigned templates, when generating duties later on in the process, the first 3 templates are assigned a night duty and the 4th one a late duty, while this is not allowed. We assume that this issue is prevented by then assigning the 4th template a night duty.

At least 1 Red Weekend needs to occur every 3 weeks, which is modelled as

$$\sum_{i \in C3_c^l} RW_i \geq 1 \qquad \forall c = 1, ..., g, \forall l \in C_c, \tag{23}$$

with $C3_c^l$ being the set of 3 consecutive rows corresponding to roster group c starting from row l and

$$t((i,5), h(i,7)) \geq 3600 RW_i \qquad \forall i \in C \tag{24}$$

setting $RW_i = 0$ if the time between the templates on Friday and Monday does not exceed 3600 minutes. As advised by planners of NS, Saturday and Sunday of a Red Weekend need to be days of type R:

$$x_{ij}^R \geq RW_i \qquad \forall i \in C, j \in \{6, 7\}, \tag{25}$$

which sets $RW_i = 0$ whenever at least one of those days is not assigned R. The last constraint

$$\sum_{b \in B_5} x_i^b M_b \leq 1 - RW_i \qquad \forall i \in C \tag{26}$$

fixes $RW_i = 0$ if $b \in B_5$ with end time after 00.00 is assigned to Friday of row i.

3.2 Reducing complexity by breaking symmetries

To determine the model's number of constraints and variables, we assume $|W|$, $|D|$ and $|A|$ are all constants. Then, the number of decision variables corresponds to $\mathcal{O}(|B||C|)$. As $|B| = \sum_{j \in W} |B_j| \leq 7 \max_{j \in W} |B_j|$ and $\max_{j \in W} |B_j| \leq |C|$, the number of decision variables for a feasible instance is given by $\mathcal{O}(|C|^2)$. Furthermore, the number of constraints is given by $\mathcal{O}(|C|)$. Hence, we conclude that the model size grows with the number of crew base's members.

A large number of model solutions are symmetric, which increases the complexity of the model. As the subrosters corresponding to a single roster group are cyclic, we can move up each row by one and obtain the same roster. However, regular, naive branch and cut algorithms ignore this and spend time on unnecessary branches, which result in identical solutions. To decrease the naive solution space, caused by these symmetry issues, we introduce extra constraints. For each roster group $c = 1, ..., g$, we propose the following:

$$RW_{C_c^1} = 1, \tag{27}$$

which forces that each subroster's first row is assigned a Red Weekend. Here, C_c^1 is the smallest element of C_c: the first row corresponding to roster group c. As at least one out of every three weekends is a Red Weekend (see Table 1), we can remove $\leq \frac{2}{3}$ of the symmetric solutions corresponding to moving up each row within a single subroster.

Table 3: Characteristics of the crew bases used.

Crew Base	#Templates per week	#Crew members	#Roster groups
Enkhuizen	147	34	2
Nijmegen	342	82	8
Utrecht	896	218	25

Table 4: Model results of the feasibility problem. In *italics*, the results after including symmetry breaking are given.

Crew Base	#Variables	#Constraints		#Nodes		Time (s)	
Enkhuizen	8126	7597	*7599*	2169	*1081*	30	*22*
Nijmegen	35588	28506	*28514*	6179	*7403*	268	*211*
Utrecht	215384	111887	*111912*	19263	*14969*	11791	*5758*

4 Real-world validation

The NS provided real-world data to answer the feasibility question posed. Here, we present how this data was used to answer this question and validate the model. To solve the ILP for the NS rosters, the MIP solver IBM ILOG CPLEX Optimization Studio v22.1.1 was used, using default parameter settings.

4.1 Feasibility: Aligning templates

The NS templates used as input are generated for train drivers per crew base for a whole week. We have taken Dutch crew bases of varying sizes: Enkhuizen, Nijmegen, and Utrecht (see Table 3), which are both representative and feasible, with Utrecht having the Netherlands' largest crew base.

To determine the feasibility of making rosters using templates instead of duties, we verified whether or not all templates can be assigned to the rosters. As the requirements were modelled as hard constraints in an ILP , its solutions found are guaranteed to be feasible.

Table 4 shows that all instances are solved. So, it is indeed possible to generate feasible rosters, using templates instead of duties. Following Section 3.2, the model size (i.e., the number of variables and constraints) grows with the number of crew members in the instance. This also holds for the number of nodes explored in the branch and bound tree and the time needed to solve the model. Considering the resulting rosters, a striking feature is that the days off are not evenly spread over the weeks (see Figure 2).

4.2 Adding symmetry breaking constraints

To prevent the solver from searching unnecessary branches, resulting in identical solutions, we added symmetry breaking constraints to our model (see (27)). This reduced the computation time for all instances, up to even 50% for the largest instance, see Table 4. Given the reduced computation time for all instances, we adopt the symmetry breaking constraints in our model.

	Mo	Tu	We	Th	Fr	Sa	Su	**Total**
1		×		×		×	×	**4**
2		×	×	×		×	×	**5**
3	×			×	×			**3**
4		×	×	×		×		**4**
5		×	×	×	×		×	**5**
6			×		×			**2**

	Mo	Tu	We	Th	Fr	Sa	Su	**Total**
1	×	×	×	×				**4**
2			×			×	×	**3**
3		×		×		×	×	**4**
4	×	×		×	×			**4**
5	×		×	×			×	**4**
6		×	×			×	×	**4**

Fig. 2: Roster for a group of 6 members working part time at crew base Utrecht as solution to the feasibility problem (left) and the optimization problem (right). The crosses indicate the assignment of templates.

4.3 Balancing Workload

For validation of the rosters created in the experiments above, we asked a roster expert within NS to evaluate their quality. As the manually made rosters used currently contain duties instead of templates (i.e., content and working times instead of only placeholders for working times), we cannot directly compare the rosters currently in use to the ones generated here in terms of quality. From this evaluation, it turned out that the rosters realized were judged as unattractive to crew members, as among other things the workload is not well balanced over the weeks. From this qualitative feedback, we propose to adjust the model in the following manner to obtain more attractive rosters: We add soft constraints, which allows constraints to be violated against a certain cost, and aim to spread templates more evenly over the weeks. Consequently, next to finding a solution that meets all requirements (i.e., a feasibility problem), our model can be approached as an optimization problem (i.e., contains an objective). To realize this, we introduce

$$\min \sum_{i \in C} w_i \tag{28}$$

$$w_i \geq 2 \left(\sum_{j \in W} \sum_{b \in B_j} x_i^b - L_c \right) \qquad \forall c = 1, ..., g, \forall i \in C_c \tag{29}$$

$$w_i \geq 0, \text{ integer} \qquad \forall i \in C, \tag{30}$$

which allows to balance the workload over the weeks. Here, L_c is the desired average number of templates per week for roster group c: a 32 and 36 hour contract corresponds to respectively $L_c = 4$ and $L_c = 4.5$. By bounding the variables w_i by 0 from below, we only penalize whenever workload is above the desired average. This formulation has the advantage of steering towards 2 weeks of 5 templates rather than 1 week of 6 templates and 1 of 4 templates.

Table 5 shows the results when adding Equations (28), (29), and (30) to the model. For all crew bases, the found solutions are proven to be optimal with respect to the used objective (28)(i.e., the gap reported by CPLEX is 0.0%). The resulting rosters (e.g., Figure 2) are evaluated by the roster expert as more satisfactory.

Table 5: Results optimization problem including symmetry breaking constraints.

Crew Base	#Variables	#Constraints	#Nodes	#Solutions	Time (s)	Value	Gap
Enkhuizen	8160	7633	2241	6	54	11	0.0%
Nijmegen	35670	28596	8637	1	345	18	0.0%
Utrecht	215602	112130	112951	36	142676	39	0.0%

5 Discussion

To answer the feasibility question posed by NS, we have developed an integrated, tractable ILP model to solve Cyclic Crew Rostering Problems, using templates instead of duties. This allowed feasible rosters respecting labour regulations when considering real-world data. Further, computational complexity was reduced by removing symmetries in the solution space. However, an NS expert judged the resulting rosters as unattractive to its crew members. Therefore, soft constraints were introduced that allowed balancing the workload over the weeks. This provided rosters that are optimal with respect to the balancing of workload and more satisfactory to crew members.

Remarks on the proposed method can be made. As shown in Section 3.1, the roster rule *Weekly rest time* was modelled less strict than the labour agreements, which might cause a feasibility issue in practice. The practical implications of this need to be studied and, if needed, countermeasures need to be taken. Furthermore, it needs to be noted that the ILP model needs updates when rules are changed. For example, NS is planning to increase the number of Red Weekends per year. This might lead to a reduced effect of the symmetry breaking constraints (see Equation (27)). Then, other symmetry breaking constraints can be considered to reduce computation times. All in all, we proposed a model which reflects a real-world problem, taking into account all real-world constraints and not simplifying those. As such, results are realistic and thus useful for NS: they now know that it is possible to create feasible rosters using templates. They even use the model proposed above to further research the newly proposed template based crew planning process.

The rosters generated by the ILP model need to be further improved in terms of fairness [7, 19] and attractiveness [7] in order to use them in practice, as also becomes apparent from informal feedback of different NS roster experts. For example, days off are preferred consecutively and increasing starting times are desired within a sequence of templates (i.e., not to much fluctuations in starting times). Also, crew members' specific preferences can be considered. However, these additions imply adding more either hard or soft constraints to the model, which would increase the model's computational complexity. For this reason, future research should also consider different, possibly hybrid, solving methods, such as AI -induced decomposition techniques using column generation [9, 10]. Furthermore, preventing a cold start problem in solving the ILP could speed up the process, as indicated by Er-Rbib et al. [11]. Because of the nature of the complex rostering rules, it is worth considering Constraint Programming [9, 10] to generate rosters.

In sum, an ILP model is presented. This allows to generate constraint and template-based, integrated, exact, and tractable models for cyclic crew rosters. However, these rosters show to be unsatisfactory for the crew members they would be assigned to. To satisfy part of the the crew members' preferences, soft constraint are added to the models to improve workload balance, which improves the roster quality. To improve roster quality even more, AI -OR approaches are promising, uniting the best of both worlds, especially for complex, real-world scenarios, which often ask for *good* instead of *optimal* solutions.

Acknowledgments. The authors have received funding for this work from the Dutch Top Consortia for Knowledge and Innovation (TKI) of the Creative Industry (CLICKNL) for the project HUPLAN - the human central, under grant agreement no. TKI1906-CI21012. We thank Sandra Butzelaar (NS expert) for explaining the NS collective labor agreement, providing crew base compositions data, and examining the generated rosters.

Disclosure of Interests. The authors have no competing interests to declare.

Appendix

Here, we define the helper functions used in the model formulation of Section 3.

We define the function $h : C \times W \to C \times W$, which gives the cell in the subroster that corresponds to the day after cell (i, j) as:

$$h(i,j) = \begin{cases} (C_c^1, 1) & \text{if } j = 7 \text{ and } i = C_c^1 + |C_c| - 1, i \in C_c, \\ (i+1, 1) & \text{if } j = 7 \text{ and } i \neq C_c^1 + |C_c| - 1, i \in C_c, \\ (i, j+1) & \text{else,} \end{cases} \tag{31}$$

where C_c^1 is the smallest element of C_c, i.e., the first row corresponding to roster group c. We can use h recursively if we want to know what cell corresponds to n days after cell (i, j) by calling $h^n(i, j)$. Here, $h^0(i, j) = (i, j)$ and $h^1(i, j) = h(i, j)$. We denote $h^{-1}(i, j)$ for the cell before (i, j).

The function $t((i_1, j_1), (i_2, j_2)) : (C \times W) \times (C \times W) \to \mathbb{R}$, which gives the time in minutes between the end of the template on cell (i_1, j_1) end the beginning of the template on cell (i_2, j_2), or guarantees enough minutes if one of the cells is not assigned a template, is given by

$$\begin{aligned} t((i_1, j_1), (i_2, j_2)) =& 1440 - \sum_{b \in B_{j_1}} x_{i_1}^b E_b + \sum_{b \in B_{j_2}} x_{i_2}^b S_b + \sum_{d \in D} x_{i_2 j_2}^d S \\ &+ 1440 \left(j_2 + 7 \left((i_2 - i_1) \mod |C_c| \right) - j_1 - 1 \right), \end{aligned} \tag{32}$$

where $i_1, i_2 \in C_c$ and $S = \max_{b \in B} E_b$. We now have the following cases if not both cells are assigned a template:

1. A template is assigned to cell (i_2, j_2) only. Then, (32) returns a value equal to 1440 plus $\sum_{b\in B_{j_2}} x^b_{i_2} S_b$ and the amount of minutes corresponding to the number of days between the considered cells. A lower bound on this value is $1440\left(j_2 + 7\left((i_2 - i_1) \mod |C_c|\right) - j_1\right)$.
2. Both cells (i_1, j_1) and (i_2, j_2) contain a type of D. Then, (32) returns a value equal to 1440 plus $\max_{b\in B} E_b$ and the amount of minutes corresponding to the number of days between the considered cells. The same lower bound holds as in the previous case.
3. Only cell (i_1, j_1) is assigned a template. Now, it can happen that a large number is subtracted from (32) in the second term. However, it cannot be larger than what is added in the the fourth term, which equals $\max_{b\in B} E_b$. So again, a lower bound on the value that (32) returns in this case is $1440\left(j_2 + 7\left((i_2 - i_1) \mod |C_c|\right) - j_1\right)$.

All in all, if we make sure the constraints we make always hold whenever t returns at least $1440\left(j_2 + 7\left((i_2 - i_1) \mod |C_c|\right) - j_1\right)$, in practice we only consider the constraint whenever templates are assigned to both cells.

Next, we define a variant of the above function, $t^U((i_1, j_1), (i_2, j_2)) : (C \times W) \times (C \times W) \to \mathbb{R}$, in which we only care for templates $b \in B$ on (i_1, j_1) which have a certain property corresponding to the binary variable $U_b \in \{0, 1\}$ as

$$\begin{aligned} t^U((i_1, j_1), (i_2, j_2)) =& 1440 - \sum_{b\in B_{j_1}} x^b_{i_1} E_b U_b + \sum_{b\in B_{j_2}} x^b_{i_2} S_b + \sum_{d\in D} x^d_{i_2 j_2} S \\ &+ 1440\left(j_2 + 7\left((i_2 - i_1) \mod |C_c|\right) - j_1 - 1\right), \end{aligned} \tag{33}$$

where $i_1, i_2 \in C_c$.

When guaranteeing the norm of a (series of) R, S day(s), a WTV day can be seen as a template with start time 120 (02.00) and end time 1440 (24.00). This allows us to make use of a modified version of the function t in which we treat a WTV day as a template with mentioned start and end time:

$$\begin{aligned} t^{WTV}((i_1, j_1), (i_2, j_2)) =& 1440 - \sum_{b\in B_{j_1}} x^b_{i_1} E_b - 1440 x^{WTV}_{i_1 j_1} + \sum_{b\in B_{j_2}} x^b_{i_2} S_b \\ &+ \sum_{d\in \{R,S\}} x^d_{i_2 j_2} S + 120 x^{WTV}_{i_2 j_2} \\ &+ 1440\left(j_2 + 7\left((i_2 - i_1) \mod |C_c|\right) - j_1 - 1\right), \end{aligned} \tag{34}$$

where $i_1, i_2 \in C_c$.

References

1. Collectieve Arbeidsovereenkomst NS 2022–2023, URL:https://www.werkenbijns.nl/static/uploads/50229-n-s-ca-o-2020-2023-digitale-versie.pdf., (Last accessed 25 October 2024)
2. Abbink, E., Huisman, D., Kroon, L.: Railway crew management, International Series in Operations Research & Management Science, vol. 268, chap. 11, pp. 243–264. Cham, Switzerland: Springer International Publishing AG (2018)
3. Abbink, E., Fischetti, M., Kroon, L., Timmer, G., Vromans, M.: Reinventing crew scheduling at netherlands railways. Interfaces **35**(5), 393–401 (2005)
4. Aghaei, S., Gómez, A., Vayanos, P.: Strong optimal classification trees. Operations Research (in press)
5. Bengio, Y., Lodi, A., Prouvost, A.: Machine learning for combinatorial optimization: A methodological tour d'horizon. European Journal of Operational Research **290**(2), 405–421 (2021)
6. Van den Bergh, J., Beliën, J., De Bruecker, P., Demeulemeester, E., De Boeck, L.: Personnel scheduling: A literature review. European Journal of Operational Research **226**(3), 367–385 (2013)
7. Breugem, T., Dollevoet, T., Huisman, D.: Crew planning at netherlands railways: Improving fairness, attractiveness, and efficiency. In: Huisman, D., Zaroliagis, C.D. (eds.) 20th Symposium on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2020). vol. 85, pp. 7:1–7:16. GermanySchloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany, Pisa, Italy (7–8 September 2020)
8. Chancellor, S.: Toward practices for human-centered machine learning. Communications of the ACM **66**(3), 78–85 (2023)
9. Dilkina, B.: Integration of Constraint Programming, Artificial Intelligence, and Operations Research: Proceedings of the 21st International Conference (CPAIOR2024) – Part I, Lecture Notes in Computer Science (LNCS), vol. 14742. Cham, Switzerland: Springer Nature Switzerland AG (2024)
10. Dilkina, B.: Integration of Constraint Programming, Artificial Intelligence, and Operations Research: Proceedings of the 21st International Conference (CPAIOR2024) – Part II, Lecture Notes in Computer Science (LNCS), vol. 14743. Cham, Switzerland: Springer Nature Switzerland AG (2024)
11. Er-Rbib, S., Desaulniers, G., El Hallaoui, I., Bani, A.: Integrated and sequential solution methods for the cyclic bus driver rostering problem. Journal of the Operational Research Society **72**(4), 764–779 (2021)
12. Fan, Z., Ghaddar, B., Wang, X., Xing, L., Zhang, Y., Zhou, Z.: Artificial intelligence for operations research: Revolutionizing the operations research process. arXiv preprint arXiv:2401.03244 (2024)
13. Gambella, C., Ghaddar, B., Naoum-Sawaya, J.: Optimization problems for machine learning: A survey. European Journal of Operational Research **290**(3), 807–828 (2021)
14. Graziani, M., Dutkiewicz, L., Calvaresi, D., Amorim, J.P., Yordanova, K., Vered, M., Nair, R., Abreu, P.H., Blanke, T., Pulignano, V., Prior, J.O., Lauwaert, L., Reijers, W., Depeursinge, A., Andrearczyk, V., Müller, H.: A global taxonomy of interpretable AI: Unifying the terminology for the technical and social sciences. Artificial Intelligence Review **56**(4), 3473–3504 (2023)
15. Gupta, S., Modgil, S., Bhattacharyya, S., Bose, I.: Artificial intelligence for decision support systems in the field of operations research: review and future scope of research. Annals of Operations Research **308**, 215–274 (2022)

16. Hanne, T., Dornberger, R., Frey, L.: Multiobjective and preference-based decision support for rail crew rostering. In: 2009 IEEE Congress on Evolutionary Computation. pp. 990–996. IEEE (2009)
17. Hartog, A., Huisman, D., Abbink, E.J.W., Kroon, L.G.: Decision support for crew rostering at NS. Public Transport **1**(2), 121–133 (2009)
18. Kobbacy, K.A.H., Vaderaand, S., Rasmy, M.H.: AI and OR in management of operations: History and trends. Journal of the Operational Research Society **58**(1), 10–28 (2007)
19. van Rossum, B.T.C., Dollevoet, T., Huisman, D.: Railway crew planning with fairness over time. European Journal of Operational Research **318**(1), 55–70 (2024)
20. Rudin, C.: Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nature Machine Intelligence **1**(5), 206–215 (2019)
21. Scavuzzo, L., Aardal, K., Lodi, A., Yorke-Smith, N.: Machine learning augmented branch and bound for mixed integer linear programming. Mathematical Programming pp. 1–44 (2024)
22. Stevens, A., Smedt, J.D.: Explainability in process outcome prediction: Guidelines to obtain interpretable and faithful models. European Journal of Operational Research **317**(2), 317–329 (2024)
23. Taghipour, S., Namoura, H.A., Sharifi, M., Ghaleb, M.: Real-time production scheduling using a deep reinforcement learning-based multi-agent approach. INFOR: Information Systems and Operational Research **62**(2), 186–210 (2024)
24. Tsay, C., Kronqvist, J., Thebelt, A., Misener, R.: Partition-based formulations for mixed-integer optimization of trained relu neural networks. In: Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., Vaughan, J.W. (eds.) Advances in Neural Information Processing Systems. vol. 34, pp. 3068–3080. Curran Associates, Inc. (2021)
25. Vossen, T., Ball, M., Lotem, A., Nau, D.: On the use of integer programming models in ai planning. In: Aiello, L.C., Dean, T. (eds.) Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI-99). vol. 1, pp. 304–309. Cambridge, MA, USA: AAAI Press, Stockholm, Sweden (July 31 – August 6 1999)