A flexible mixed integer programming based system for real-world nurse rostering

Bodvarsdottir, Elin Bjørk; Bagger, Niels-Christian Fink; Høffner, Laura Elise; Stidsen, Thomas Jacob Riis

Published in:
Journal of Scheduling

Link to article, DOI:
10.1007/s10951-021-00705-7

Publication date:
2022

Document Version
Peer reviewed version

Journal of Scheduling manuscript No.
(will be inserted by the editor)

A flexible mixed integer programming based system for real-world nurse rostering

Elín Björk Böðvarsdóttir · Niels-Christian Fink Bagger · Laura Elise Høffner · Thomas J. R. Stidsen

Received: date / Accepted: date

Abstract Researchers have studied the nurse rostering problem (NRP) for multiple decades. Initially, the formulations were rather primitive including only a few necessary restrictions, but down the road, the formulations have become more complex. Nonetheless, a fraction of the research reaches implementation in practice, and many wards still schedule nurses manually. In this article, we introduce a flexible nurse rostering system that employs mathematical optimization to automatically schedule nurses to shifts. We have developed this system in collaboration with practitioners to fully match their needs. The system consists of a comprehensive mixed integer programming (MIP) model along with a flexible framework. In addition to common constraints from the literature, the mathematical formulation includes three new constraints that further encourage healthy work schedules for each nurse. Additionally, we have reformulated some common constraints from the literature and allow for a complex shift structure that matches the needs of real hospital wards. This flexibility results in increased adaptability for different wards with different needs and is crucial to address the complex nurse rostering problem that practitioners face. We have successfully implemented this system in two wards at two Danish hospitals. We present the MIP model along with computational results for 12 real-world rostering instances. Furthermore, we discuss the practical impact of this system and provide general feedback from the practitioners using it. Overall, the results illustrate the capabilities of the system to tackle diverse nurse rostering instances and produce outstanding results.

Keywords Nurse rostering · Automatic scheduling · Mixed Integer Programming · Preference scheduling · OR applications in healthcare

Elín Björk Böðvarsdóttir
Department of Technology, Management and Economics, Technical University of Denmark, Anker Engelundsvej 1, 2800 Kgs. Lyngby, Denmark
E-mail: ebod@dtu.dk

Niels-Christian Fink Bagger
Department of Technology, Management and Economics, Technical University of Denmark, Anker Engelundsvej 1, 2800 Kgs. Lyngby, Denmark
Fink Optimization ApS, Valdemarskrogen 44, 2860 Søborg, Denmark

Laura Elise Høffner
Department of Data and Development Support, Region Zealand, Alleen 15, 4180 Sorø, Denmark

Thomas J. R. Stidsen
Department of Technology, Management and Economics, Technical University of Denmark, Anker Engelundsvej 1, 2800 Kgs. Lyngby, Denmark

1 Introduction

The nurse rostering problem (NRP) is the problem of assigning nurses to shifts to create a roster satisfying some predetermined requirements. In academic research, the roster often consists of three work shifts a day (i.e., early, late and night shifts) and the requirements are generally divided into two categories, coverage requirements, and nurse specific requirements. This problem has been studied for decades (Burke et al., 2004; Van Den Bergh et al., 2013), resulting in a wide variety of formulations and solution algorithms for different instances.

Even though researchers have made many advances in modeling and solving variants of the NRP, we can still see a gap from academia to implementation in practice. An exploratory study by Kellogg and Walczak (2007) showed that only 30% of nurse rostering systems

from academic research were used in practice, and most only by a single ward or a single hospital. Moreover, Van Den Bergh et al. (2013) reported that out of 64 articles considering the NRP, less than 20% were applied in practice, despite 90% using real-world data. Furthermore, Petrovic (2019) recently concluded that most of the available software for personnel scheduling had not benefited from academic advances. She encouraged researches to engage more with practitioners to explore all the complex problems that arise in the real-world.

In this article, we introduce a flexible nurse rostering system that meets the requirements set by real-world hospitals. This system was developed in collaboration with experienced practitioners and has been implemented in two wards at two hospitals in Region Zealand, Denmark. We formulate the problem as a mixed integer programming (MIP) model and solve it using an exact solver. As the solver reaches a low optimality gap within a short time, we see no need to develop any heuristics for solving it. Furthermore, a MIP model is easy to maintain and extend to accommodate multiple wards or general changes in the requirements.

In addition to this model, we also present a flexible framework. As an example, the framework allows for a complex shift structure along with handling several special cases that are often omitted from academic research. Having the flexibility to meet the needs of different wards is crucial in practice, as special-tailored systems are not only costly but also difficult to maintain, thus being unlikely to succeed in the long run.

When solving the NRP we need to consider multiple stakeholders with different interests, one of the largest stakeholders being the nurses themselves. Gärtner et al. (2018) emphasized the importance of producing good rosters by addressing multiple aspects including health, safety, social well-being, and work-life balance. As poor rosters can have a significant impact on the personnel's quality of life, working with human resources becomes much more complex than scheduling other types of resources.

While *cyclical scheduling* (i.e., where the same roster is repeated over multiple rostering horizons) was common in early NRP studies (Baker, 1974; Burns and Koop, 1987), it lacks the flexibility to fully meet the nurses' needs. Nowadays the rosters are generally tailored to each rostering horizon, giving the nurses an opportunity to influence their rosters. Two common methods are *self-scheduling* and *preference scheduling*. In *self-scheduling* the nurses cooperate to generate a roster by signing up to shifts and jointly solving conflicts. In *preference scheduling* the nurses may request shifts and days-off but the personnel manager is responsible for solving conflicts and building a final schedule around said requests (Bard and Purnomo, 2005). When generating unique rosters for each horizon, past assignments need to be considered to ensure feasibility across the border of adjacent rostering horizons (Glass and Knight, 2010; Smet et al., 2017).

Although formulations of the NRP may differ, the constraints that they include are often similar, e.g., ensuring sufficient rest between shifts, limiting consecutive work days and meeting a pre-defined staffing requirement. Most often, the constraints are categorized as hard or soft constraints and the objective is to minimize the violation of soft constraints. Although far from ubiquitous, the coverage constraints are often hard while nurse-specific constraints are soft. De Causmaecker and Vanden Berghe (2011) introduced an $\alpha|\beta|\gamma$-notation for classifying nurse rostering problems, providing researchers with a basis to compare and contrast various NRP studies.

The hardness of the nurse rostering problem can vary significantly depending on which constraints are included and Smet et al. (2016) showed that some variants of personnel rostering problems were polynomially solvable. Despite that, one of their results was that adding specific constraints on consecutive days to a polynomially solvable problem, made the resulting problem NP-hard. Unfortunately, these types of constraints frequently occur in practical problems and we can assume that formulations matching the real-world complexity will be NP-hard.

As the problem we address originates in Danish hospitals, we have included some legally binding constraints that do not appear in previous academic formulations. Even though the base is the Working Time Directive of the European Union (EU), some aspects have been adjusted in the Danish legislation. For example, weekly rest is subject to tighter constraints, making them more complex than elsewhere in the EU. We have parameterized these constraints in the MIP formulation, making them easily adjustable to fit other regulations.

A far majority of hospitals in Denmark still apply manual rostering based on rules of thumb. Prior to this project, the only exception was in Sydvestjysk hospital where they use mathematical optimization to automatically generate rosters for some wards (Christiansen et al., 2014). Manual roster generation is tremendously time-consuming and Jensen et al. (2008) presented a conservative estimate stating that the generation of rosters for nurses in Danish hospitals required 224 full-time equivalent health care employees annually. Furthermore, the resulting rosters are often of low quality and the National Audit Office of Denmark has criticized the poor utilization of personnel resources in Dan-

ish hospitals, for example pointing out a lack of transparency (Rigsrevisionen, 2015).

The major contribution of this paper is a nurse rostering system that is not only comprehensive but also flexible. The system allows for defining a broad range of shifts, tailored to the needs of each individual ward. Moreover, we embrace the uniqueness of different nurses by adapting different types of constraints to each individual. Additionally, we present a compact MIP model that can be tackled with a commercial solver within a reasonable running time. The MIP formulation includes three constraints that have never been addressed in the literature and we present more flexible alternatives for common constraints, including coverage constraints, scheduling days off and ensuring sufficient rest between work assignments. Finally, we have successfully implemented this work in practice.

The structure of the paper is as follows: Section 2 reviews the relevant literature, focusing on formulations of the NRP. Section 3 describes the problem and Section 4 introduces the MIP formulation. Section 5 presents the results, both computational and practical, and Section 6 concludes the paper.

2 Literature review

Although the NRP has been studied for decades, Burke et al. (2004) revealed that many of the problem formulations were simplified and insufficient to meet the requirements of the real world.

In 2010, Haspeslagh et al. (2014) introduced the First International Nurse Rostering Competition (INRC-I). The competition stimulated researchers and engaged them to work with several constraints encountered in practice. Furthermore, it provided a basis for comparing solution methods. While the problem formulation was more extensive than many before, Haspeslagh et al. (2014) acknowledged that it did not capture all aspects of real-world problems.

Valouxis et al. (2012), the winners of INRC-I, designed a two-phase approach to address the problem. They partitioned each instance into smaller sub-problems and solved them sequentially with an open-source integer programming solver. The first phase assigned nurses to workdays and the second phase specified the shifts worked on each day. They supplemented with heuristics that searched for improving solutions across a combination of partial schedules.

Burke and Curtois (2014) employed a branch and price algorithm to the INRC-I instances and a wide range of other benchmark instances. They presented a general rostering model where numerous common constraints were captured using regular expressions (i.e., pattern or string matching). They formulated the master problem as a set covering problem and the pricing problems as resource constrained shortest path problems. They used dynamic programming along with some heuristic rules to solve the pricing problems. They compared their results to those found with an ejection chain based approach.

Curtois and Qu (2014) employed these methods introduced by Burke and Curtois (2014) to a set of new benchmark instances. While these instances included some common constraints from the literature, the number of different types had been reduced to make the instances easier to use and test. They presented 24 instances that varied in size and complexity. Furthermore, Curtois and Qu (2014) presented a compact MIP formulation of the problem and solved it using Gurobi.

In 2015, Ceschia et al. (2019) introduced the Second International Nurse Rostering Competition (INRC-II). They proposed a multi-stage formulation, requiring data to be carried between stages (i.e., weeks) to ensure an overall feasible roster. In that manner, the competition addressed a drawback in previous research, which had often considered the NRP as an isolated rostering horizon.

Römer and Mellouli (2016), the winners of INRC-II, employed a network flow-based MIP formulation to address the problem. For each nurse, they designed a directed acyclic network layer with arcs for shifts and days off. They extended the network with states that captured the information needed to evaluate most of the soft constraints. This approach resulted in a large but strong formulation, which they solved with an open-source solver. To reduce its size, Römer and Mellouli (2016) heuristically removed certain nodes and arcs that were unlikely to result in a good solution.

Mischek and Musliu (2017) presented a compact MIP model for the INRC-II formulation. They also introduced some extensions (i.e., additional constraints) to reduce the negative impact of incomplete information related to multiple stages. They solved their extended formulation with CPLEX and concluded that it was competitive to the finalists of the INRC-II.

Several researchers have successfully implemented nurse rostering solution methods in practice. Many of those systems are based on meta-heuristics (Bilgin et al., 2012; Burke et al., 2001) or hybridizing meta-heuristics with other approaches, such as artificial intelligence (Beddoe et al., 2009) or constraint programming (Stølevik et al., 2011).

Due to the hardness of the problem, exact methods were not widely used in the early years. Nevertheless, with improvements in computational and algorithmic power, such methods have become promisable. In re-

cent years, several researchers have hybridized integer programming (IP) with heuristics (Burke et al., 2010; Rahimian et al., 2017; Turhan and Bilgen, 2020) and others have presented exact solution approaches based on decomposing the problem (Maenhout and Vanhoucke, 2010; Dohn and Mason, 2013).

Santos et al. (2016) employed IP techniques to the INRC-I formulation. They improved the lower bound by adding strong inequalities based on the problem structure, namely clique cuts, and odd-hole cuts. Furthermore, they formulated the problem in terms of working windows and resting windows that should alternate throughout the rostering horizon. Drawing upon this alternating structure, they introduced additional cuts that further strengthen the formulation. They introduced a MIP heuristic to speed up the generation of near-optimal solutions. With this approach, they proved optimality for a vast majority of the INRC-I instances and improved best-known solutions for other instances.

A few researchers have employed compact MIP models to address the NRP. These models are often quite simplified, for example, formulated around a pre-defined shift pattern (e.g., three fixed shifts a day) and even assuming homogeneous workforce (Agyei et al., 2015; Azaiez and Al Sharif, 2005; Zanda et al., 2018). Such models lack the flexibility needed to fully meet the needs of different wards along with the individual needs of different nurses.

De Grano et al. (2009) introduced an IP model for incorporating nurses' direct requests with the use of an auction. In this auction, each nurse was allocated the same number of points to bid for shifts and days off, providing them with control over the importance of different requests. They focus on maximizing the score for the winners subject to a few scheduling rules, such as sufficient rest.

Lin et al. (2014) presented an IP model that utilized preference ranks when generating nurse rosters. A nurse could classify each shift as being good, normal, or bad, and similarly classify possible days-off as good or bad. The objective was to maximize the weighted preference satisfaction. A large limitation of this model was that they assume that nurse is always assigned to the same shift throughout the rostering horizon, i.e., forbidding rotations, which substantially reduces the number of variables and constraints. Lin et al. (2015) extended this work by allowing a nurse to be assigned to different shifts. Furthermore, they modified their objective function to account for fairness when assigning preferences. They solved the extended model by employing a meta-heuristic.

Rönnberg and Larsson (2010) introduced a MIP formulation for generating rosters where the nurses should request full schedules. In addition to scheduling rules (such as sufficient rest), they also included a few quality aspects and emphasized that those are just as important as the hard rules. Their objective was to satisfy the staffing requirements with minimal deviations from the requested schedules. When measuring the fulfillment of requests they used a normalized score allowing them to objectively compare different nurses. Rönnberg et al. (2013) generalized this model, allowing other objectives and removing the requirement to request full schedules. To solve that formulation, they employed a meta-heuristic.

While researchers have employed MIP formulations and exact solvers to address the NRP, those formulations are often limited in scope. When extending these formulations to further meet the requirements of the real world, researchers have generally switched to heuristics or hybridized methods.

3 Problem description

In this section, we describe the nurse rostering formulation that we address. We have developed this formulation in dialogue with experienced managers that today use it to automatically generate rosters. The overall goal with this work has been to develop a rostering system that matches the complex needs of practitioners and to ensure its success, the system has been continuously improved using detailed feedback from the practitioners. The system consists of two components: First, a mathematical model (see Section 4), which includes all constraints required by the practitioners, and second, a framework around the model, ensuring its flexibility and applicability in practice. This structure is illustrated with Figure 1. The different terms presented in the figure will be discussed throughout the article.

The model consists of numerous hard and soft constraints. While the hard constraints enforce the most critical restrictions, the soft constraints further promote the quality of the roster and the objective function penalizes their violation. Although the NRP is multi-objective by nature, we formulate it with a single weighted-sum objective function, which is consistent with the majority of previous NRP formulations (Mihaylov et al., 2016). The division of constraints into hard and soft has been conducted in close collaboration with the practitioners. Furthermore, when tuning the weights associated with different soft constraint we focus on producing rosters matching the practitioners' demands and preferences.

Many of the constraints we include relate to common constraints from previous nurse rostering research,

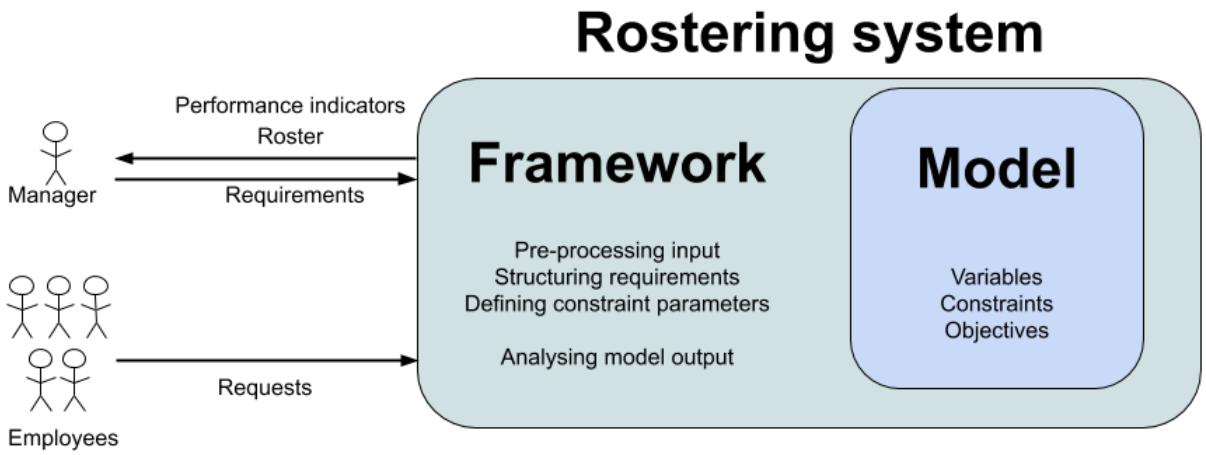

Fig. 1 The structure of the rostering system.

for example regarding backward rotations and consecutive work assignments. Nontheless, we formulate some of these constraints in a more general manner, to ensure the overall flexibility of the system. In addition to common constraints, we introduce some new constraints (see Section 3.2.9), which were included based on feedback from practitioners. Overall, the problem we address is NP-hard.

This formulation provides several extensions to previous nurse rostering formulations. These extensions all aim to increase the flexibility, resulting in a generalized formulation that should be applicable to real-world problems with different characteristics. The most important extensions are as follows:

- The incorporation of different types of days off and their definition as shift types.
- The definition of shift sets that group related shifts.
- An alternative formulation of constraints restricting the work on a single day or the rotation of work on two adjacent days.
- A flexible formulation of coverage constraints that allows for matching different shifts that provide partial coverage.
- The possibility of assigning nurses to shifts that do not correspond to any coverage requirements, without these assignments being pre-defined.

This problem categorizes as $ASBCI|RVNO|PLX$ according to the $\alpha|\beta|\gamma$-notation introduced by De Causmaecker and Vanden Berghe (2011). For the β-category (i.e., coverage constraints), the formulation does not fully belong to the T subcategory, which relates to coverage constraints over time intervals as opposed to shifts. Nonetheless, the flexibility of the formulation allows for satisfying a single coverage constraint by matching different shifts (see Section 3.2.6), and thus we argue that this formulation is on the verge of the T-subcategory.

In the following sections, we will present the rostering system in detail, and link it to previous research as relevant. Section 3.1 describes the framework around the model, while Section 3.2-3.3 describe the constraints and objective, respectively. As hospitals in Denmark do not only employ nurses, but also other caring professions that contribute towards the staffing requirements, we will use the general term *employees* throughout this paper. Working with different caring professions results in a broader set of skills, which need to be managed with the coverage constraints.

3.1 General framework

The framework around the model is flexible, where all aspects are user-definable and thus applicable to various wards with different structures and different needs. By incorporating data from the previous rostering horizon, we ensure feasibility across rostering horizons without explicitly modeling the border. Furthermore, the managers can easily modify the input to account for the needs of each ward, e.g., by lowering the staffing requirements around holidays or updating individual skills, even in the midst of a rostering horizon.

The main goal of this framework is to manage the underlying complexity of the problem. It keeps unnecessary details outside the optimization model, without conducting simplifications that impact the applicability of the system. We present a general optimization model that can be customized to fit the needs of each ward (or each employee) by modifying parameters in the framework.

In the Sections 3.1.1-3.1.2, we will discuss two aspects of the framework in more detail, namely the definition of *shifts* along with how we promote the *individuality* of each employee. The flexibility of these aspects is essential for implementing automatic roster generation in practice.

3.1.1 Shifts

In previous research, the term *shift* has generally referred to work shifts, often fixed as day, evening and night. INRC-I and INRC-II defined a shift type as "a time frame for which a nurse with a certain skill is required" (Ceschia et al., 2019; Haspeslagh et al., 2014), i.e., defining shifts around the staffing requirements. While other researchers have included assignments besides pure staffing requirements, they have done so explicitly (Rönnberg and Larsson, 2010).

In this paper, we extend the definition of shifts, stating that *a shift type is a time frame corresponding to any event that we can assign to an employee.* This extension implicitly includes work shifts that do not correspond to any coverage constraints (e.g., administrative or educational shifts). Furthermore, it includes days off as shift types.

Although the inclusion of all work-shifts comes natural, the inclusion of days off as shift types can be debatable. Nonetheless, we argue that viewing days off as shift types can be highly beneficial in a practical setting. Even though academic research may view all days off as the same, in practice, these days can be of different types where the circumstances for scheduling them also differ. Examples include days for annual leave and parental leave, along with legally binding days off for resting (which we refer to as a *protected days off* in this paper).

Moreover, some types of days off contribute towards the overall number of hours we should assign to an employee. As an example, by assigning an employee to two weeks of annual leave we consequently reduce the number of working hours we should assign to him or her during the rostering horizon.

With this extended definition, we introduce a constraint stating that we should assign at least one shift type each day. We acknowledge that this constraint contradicts a common constraint from previous research, namely allowing at most one shift type a day, and we will provide a thorough comparison in Section 3.2.1. To ensure the feasibility of this new constraint, we define a dummy shift type *Free* for a day off that does not match any specific category.

In addition to shift types, we also define shift sets as follows: *A shift set is a set of shift types where each shift type in the set has an associated parameter denoting the relationship between the shift type and the shift set.* Figure 2 presents an example with multiple shift types and how they relate to different shift sets.

In this figure, shift types *D1*, *D2* and *D3* all belong to the *Day* shift set, and assigning an employee to any of these shift types contributes towards a coverage requirement corresponding to that shift set. The same holds for shift types *E1* and *E2* for the *Evening* shift set, and shift types *N1* and *N2* for the *Night* shift set.

We also include a shift set *DayAll*, which has *Day* as a subset. The reason we include this extension is that we may assign some employees to work during this timeframe without them contributing towards any coverage constraints (for example the administrative shift *ADM*). Even though they do not contribute to any coverage we still need to satisfy multiple employee-specific constraints, e.g. ensuring sufficient rest between shifts. In theory, we could also have similar extensions for the *Evening* and *Night* shift sets, but in practice we generally assign this type of work during the day, thus minimizing additional salary expenditures.

In addition to work, we also include the shift set for *Off*, which includes two shift types: *Free* and *PF* (i.e., the shift type for a protected day off). These shift types span the entire day, from midnight to midnight, ensuring that no other shift type can be assigned simultaneously. In Section 3.2.2, we discuss the difference between these shift types and present different constraints relating to days off.

At last, we include the *DE* shift type, spanning the entire *Day* shift set along with the first half of the *Evening* shift set, and the *EN* shift type, spanning the entire *Night* shift set along with the second half of the *Evening* shift set. Thus, the parameter linking *DE* and *EN* to the *Evening* shift set is 0.5, while all other combinations of shift types and shift sets presented on the figure have the parameter as 1. In Section 3.2.6, we will discuss the increased flexibility brought by allowing this type of partial coverage.

We acknowledge that the notion of grouping shifts together is not completely new to the literature, as Bilgin et al. (2012) worked with "a set of compatible shift types". Each set corresponded to a coverage requirement (e.g., a set of night shifts) and overlap between these sets was not allowed. In addition to the coverage constraints, some nurse-specific constraints were also expressed in terms of shift sets. The extension we introduce, by allowing a shift type to partly cover a shift set, has never been presented before.

Despite the number of shift types shown in Figure 2 being large when compared to most previous research, the number of shift types used in practice is

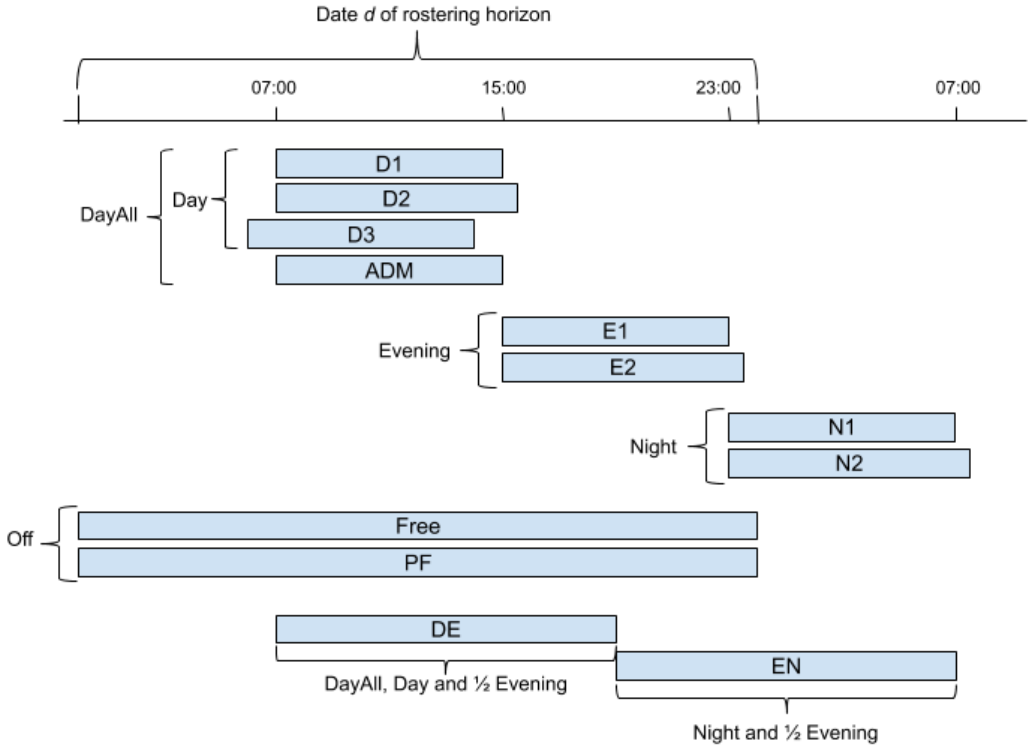

Fig. 2 Example of how different shift types are linked to shift sets. The figure presents day, evening and night shifts of different duration, along with shift types corresponding to days off. Furthermore, shift types *DE* and *EN* partly cover the *Evening* shift set, providing for increased flexibility in practice.

even greater. When providing examples throughout the paper, we will refer back to these shift types. In the rostering system, shift types and shift sets are often interchangeable, and we will generally use the term *shift* in those cases.

3.1.2 Individuality

Each employee has an individual contract that constrains his or her work. As an example, one employee might have a constraint allowing at most three night shifts in a row, while this maximum could be one for another employee. Furthermore, the corresponding constraint may be inactive for some other employees, allowing them to have many night shifts in a row as long as it is feasible w.r.t. other constraints.

We do not classify employees as full-time or part-time employees, but instead, consider the *contractual hours* of each employee, as part-time employees can have contracts ranging from 16 to 35 hours a week, while full-time employees have contracts for 37 hours a week.

Previous research generally defines some categories for skills, e.g., regular nurse or head nurse. In addition to including general categories, the system also permits individual skill definitions embedded in each shift type. This flexibility allows for including some specific considerations that managers may take into account to ensure that the employees can maintain a healthy work-life balance. As an example, an employee might have difficulties working the earliest day shift (*D3* on Figure 2) due to a long commute from home, while another employee may need to pick up children after school, thus not being suited for the long day shift (*D2* on Figure 2).

The system includes a *request environment* where the employees can make requests specific to each day of the rostering horizon. The requests can either be towards a shift (including those for days off) or against a shift (e.g., not working the night shift). Furthermore, an employee can categorize a request as either a high or low priority, corresponding to the effort that should be put into accommodating it. Assigning a requested shift is associated with a reward, and similarly assigning a shift that has a request against it is associated with a penalty.

We have not put any restrictions on how many requests the employees can make, which is common in previous research. The main reason is that we want to provide the employees with autonomy while simultaneously trusting that they will keep the interests of the

entire ward in mind. As an example, if some employees are willing to work unpopular shifts on specific days, then they can state that as requests without it restricting their ability to request some other days off. If we were to restrict the number of requests, we would punish employees for trying to meet the needs of the ward, resulting in these type of requests becoming rare.

The main drawback of not restricting the requests, is that different employees will behave differently. Some employees may be greedy and put forward an excessive amount of requests, while other are more modest and only make a few requests. We take this difference into account when allocating a reward to each request, resulting in a fair distribution of fulfilled requests.

The manager can also use the request environment to fix certain assignments beforehand, for example, due to annual leave or administrative meetings. Entering fixed assignments ensures that the optimization will not contradict any prior agreements, but instead work around them. This flexibility is of the utmost importance in practice, where employees may have other obligations preventing them from working at certain times.

To ensure that an employee's individual opinion is superior to general rules, the specific requests can override some other constraints. As an example, we generally penalize isolated work shifts (i.e., work shifts where both of the adjacent days are days off), but if an employee has specifically requested such a pattern we disregard the penalty for that specific case. Nonetheless, some constraints are not affected by the requests, e.g., we strive to assign the employees to shifts according to their contractual hours, even if they have requested an abundance of days off.

In addition to the specific requests, each employee may have some general preference for shifts, along with non-preferences, which are taken into account on days where the employee has not made a specific request. As with requests, we reward for meeting a preference and penalize for going directly against non-preferences. To promote fairness, we have normalized the reward for requests and preferences to ensure that the overall reward potential is the same across the employees, independent of the number of requests. This normalization matches the one Rönnberg and Larsson (2010) presented for requests. These rewards and penalties are included as objective weights in the formulation.

Due to the request framework, this research can be categorized as preference scheduling. Nonetheless, we add a level of complexity by including general preferences, independent of the day, and this combination is not present in any previous research.

We divide the employees into two categories: First, the employees for whom we should create a full roster, and second, the employees that we should only assign to shift in some specific cases. A majority of the employees belong to the former category, whereas the latter category includes employees that mostly work administrative shifts or belong to a subset of employees that use the request environment to fully fix their roster, i.e., true self-scheduling.

3.2 Constraints

This section describes all the constraints of the problem, categorized into nine subsections. Sections 3.2.1-3.2.5 present constraints for individual employees, while Sections 3.2.6-3.2.8 present constraints on ward level. In addition, Section 3.2.9 presents three employee-specific constraints, that have not been included in any previous research.

3.2.1 Physically infeasible or illegal combinations

In this section, we discuss pairs of assignments where scheduling one makes the other either physically infeasible or illegal. An example of a physically infeasible pair of assignments would be assigning an employee to both *D1* and *PF* on the same day, while an illegal pair would be assigning both *D1* and *E1* (as it leads to insufficient rest). We will refer to these types of assignments as *conflicting pairs*. In general, all assignments that overlap in time are conflicting, except for assigning a night shift followed by the *Free* shift type. This exception is necessary to ensure that the constraint requiring a minimum of one shift type each day is always feasible.

According to the EU's Working Time Directive, all employees should get a minimum of 11 consecutive hours off within every 24 hours. When working with three 8-hour shifts, as common in previous research, this requirement translates into a constraint allowing at most one shift a day along with a constraint forbidding backward rotations. However, according to the Danish legislation, the minimum rest of 11 hours may be reduced down to 8 hours. In general, this reduction is only allowed if the employees request it, and the frequency is substantially restricted to avoid misuse.

Therefore, a global hard constraint allowing at most one shift a day is too strict, as scheduling *D1* and *N1* on the same day may be feasible. The same holds for forbidding backward rotations, as scheduling *E1* on one day and *D1* on the next may also be feasible. Nonetheless, the feasibility of these pairs depends on both the employee and the day. For example, employee e_1 may request *D1* and *N1* on day d_1. Then the pair is feasible for that specific employee and day, but not for other employees on day d_1, nor on other days for employee e_1.

Additionally, the manager might prefer to assign the longer day shift *D2* and only reserve *D1* for reduced rest. Thus, constraints forbidding conflicting pairs need to be supremely flexible.

To reduce the size of the model by avoiding the enumeration of all conflicting pairs, we generate a set of *conflict cliques*, where each clique is a set of assignments such that all pairs in the set are conflicting. Section 4.1 further describes the generation of these cliques.

We note that a given clique may allow at most one shift on a given day, corresponding to the general constraint from the literature. Similarly, a given clique may forbid backward rotations. Nevertheless, the automatic generation of conflict cliques ensures that these constraints are only enforced when applicable.

3.2.2 Ensuring days off

The formulation includes some legally binding constraints from Denmark. Nonetheless, we emphasize that the formulation of these constraints is flexible, ensuring that they can be adjusted to legislation in different countries.

According to the EU's Working Time Directive, employees in EU countries are entitled to a minimum rest of 24 uninterrupted hours within every seven days. In Denmark, the rules are more favorable to the employees, stating that they should get a certain number of *protected days off* (i.e., the *PF* shift type) during a rostering horizon.

Not all days off can be considered as a protected day off, as the legislation requires a minimum number of uninterrupted hours off. For a single protected day off, the law generally requires a minimum of 35 hours. Nonetheless, the law includes the flexibility to reduce this to 32 hours, if specifically agreed with the employee. For k consecutive protected days off, where $k > 1$, the minimum number of hours is $24 \cdot k + 7 \cdot \lfloor k/2 \rfloor$. Table 1 presents some examples of the minimum hours required for a different number of protected days off.

Table 1 Minimum number of uninterrupted hours required for i consecutive protected days off. As consecutive protected days off require less hours compared to multiple isolated days, we are inclined to assign consecutive days.

i	hours
1	35 or 32
2	55
3	79
4	110

For increased apprehension of protected days off, Table 2 displays some examples where we assign either *PF* or *Free*, depending on the shifts assigned on adjacent days. The first row represents an assignment where a protected day off is infeasible, as the number of hours off is too low. The next two rows show examples of assigning protected days off between shifts, both a single day and two consecutive days. Finally, the last row shows an example where we do satisfy the minimum number of uninterrupted hours for two protected days off. However, as a night shift assigned on day d overlaps with day $d + 1$, we cannot assign the *PF* shift type on $d+1$, and thus we only assign a single protected day off. The last type of constraints are enforced using general patterns (presented in Section 3.2.5).

Table 2 Examples of assigning days off, using the shift types defined in Figure 2.

d	$d+1$	$d+2$	$d+3$
A2	**Free**	D1	
A2	**PF**	A2	
A2	**PF**	**PF**	D1
N2	**Free**	**PF**	A2

On average, employees should get two protected days off per week for the entire rostering horizon. We strive to distribute these days evenly, such that we have at most six consecutive days without assigning the *PF* shift type. Nonetheless, this distribution is a soft constraint, because the legislation allows for exceptions.

We acknowledge that the formulation does not include constraints for the number of consecutive days off, as common in previous research (Ceschia et al., 2019; Haspeslagh et al., 2014). Nonetheless, we require a total number of protected days off. Due to the number of uninterrupted hours being relatively lower for consecutive protected days off compared to a single one, we often assign them on consecutively. Thus, a constraint for a minimum number of consecutive protected days off is implicitly supported by the legally binding minimum for uninterrupted hours. A constraint for a maximum number of consecutive protected days off is supported by the requirements for even distribution along with a fixed number of protected days off for the rostering horizon. Furthermore, due to days off being categorized as shift types, we may also use general constraints for restricting consecutive assignments (presented in Section 3.2.4) to ensure a maximum on the consecutive days off.

In addition to protected days off, we must consider several other types of days off, such as *annual leave* and *parental leave*. In Denmark, the employees can generally negotiate their annual leave beforehand, and often it is fixed several months in advance. Thus, the model

should not assign the annual leave, but still take it into account when assigning other employees to shifts, e.g., ensuring that the overall staffing requirements are met.

As the legislation regarding parental leave in Denmark is both extensive and flexible, the employees' rights for parental leave can impact the roster. Employees can distribute their parental leave over a long time and may return to work while still taking partial parental leave. As an example, an employee might be entitled to one day of parental leave every week, and this day should be assigned with the overall staffing requirements of the ward in mind. Finally, it goes without saying, that the employees' rights for both annual leave and parental leave differ, not only between employees but also between rostering horizons.

The complexity of the legislation clearly displays how important it is to develop flexible systems for nurse rostering in practice. The intricacy of the different types of days off that we can encounter in practice is much greater than most academic research has acknowledged. The MIP model (see electronic appendix) embraces the inclusion of days off as shift types, as it can enforce various restrictions for days off by employing general constraints for shifts. Furthermore, the formulation for protected days off can be parameterized (i.e., modifying the values in Table 1) to suit legislation in other countries, and thus, has advantages in a global context.

3.2.3 Restricting the overall work

As in all nurse rostering formulations, we include some constraints to restrict the overall work. INRC-I and INRC-II included soft constraints stating that the number of working days for each nurse should be between a minimum and a maximum (Ceschia et al., 2019; Haspeslagh et al., 2014).

In this work, we include a similar soft constraint, but instead of counting the number of days we measure the number of hours, as shift types may differ in length. We note that some days off (e.g., annual leave) contribute towards the number of hours, along with all types of work shifts. We rarely assign exactly the number of hours we should, and use a two-factor penalty where a deviation within a certain threshold is only penalized lightly. On the contrary, a deviation beyond the threshold is penalized heavily.

An employee's individual contract specifies *contractual hours*, i.e., the number of hours that we should assign to the employee during a reference period. As the reference period generally spans multiple rostering horizons, the constraint is a global constraint, and we adjust the bounds to correct for any surplus or deficit from previous horizons. We do not account for any possible abnormalities in the upcoming horizon but assume that we will assign the employees relative to their contractual hours through the rest of the reference period.

For each employee, we divide their contractual hours into weekly targets. Although most employees would like to have the hours balanced evenly throughout the rostering horizon, we may see some exceptions where the employees wish to have the workload skewed. A skewed distribution is common for employees that have a weekly arrangement with their children, as they can spend more quality time with their children by working more in other weeks. We note that the system adjusts any general weekly targets to ensure that they match the specifically requested work time.

We include three hard constraints to manage the assignments to different types of shifts. First, an upper bound for assigning specific shifts, which is often employed to limit evening or night work. Second, a fixed number of assignments to specific shifts, for example, ensuring the correct number of protected days off. Third, a fixed number of assignments to specific shifts during a single week of the rostering horizon.

The first two constraints consider the rostering horizon as a whole, while the third is specific to each week. That constraint is used for assignments that should be distributed according to a pre-defined schedule, for example, administrative shifts or parental leave. The inclusion of these constraints, both regarding the specific shifts and the bounds, depends on the individual employee.

3.2.4 Consecutive work assignments

In practice, restricting the overall work does not suffice, and we also need to restrict consecutive work. For the number of consecutive workdays, we include constraints for both a minimum and a maximum. Furthermore, we include tighter upper bounds for some shifts. As an example, even though we may assign work on five consecutive days, an employee might have a maximum restriction for three consecutive night shifts.

Besides these constraints, INCR-II also includes a general constraint for a minimum of consecutive assignments to a given shift (Ceschia et al., 2019). We do not include that constraint in this work, except in the specific case of complete work weekends (see Section 3.2.5).

At last, we also include a constraint restricting the maximum number of hours that we can assign to an employee on a set of consecutive days, e.g., during one week. This constraint ensures that we do not solely assign an employee to long stretches of long shifts with minimal rest (e.g., one protected day off) in between. The constraint is not common in previous work, but its

importance is highlighted by Gärtner et al. (2018) including it as one of their suggestions for future rostering competitions.

3.2.5 Patterns

The formulation includes two general types of constraints for patterns, which can be customized to different wards as required.

The first type of constraints restricts unwanted series of assignments from the roster, and Table 3 presents some examples. This constraint allows for immense flexibility as the unwanted series can be tailored to the ward's needs. The constraint can either be active for the rostering horizon as a whole or only for a subset of days. Furthermore, the user controls whether each unwanted series should be strictly forbidden or penalized (along with the value of the penalty), and whether a specific request for the series should deactivate the constraint. At last, the framework does not restrict the length of the series and theoretically, the user could add a constraint with assignments spanning multiple rostering horizons.

Table 3 Examples of unwanted series. The first two present isolated work shifts, while the remaining for present unwanted shift successions (e.g., due to insufficient rest).

d	$d+1$	$d+2$
Free	Evening	Free
Free	Night	Free
Night	Free	Day
Night	Free	Evening
Night	Evening	
Night	PF	

The second type of constraints ensures that some assignments are only scheduled in a combination with other assignments. These assignments do not have to be close in time, and theoretically, we could enforce that an employee working on a Monday night should also work on Tuesday evening two weeks later. However, these type of constraints are generally enforced more locally and are for example used to ensure complete work weekends. When employing these constraints for assignments on consecutive days, they are a reverse of the constraints for unwanted series.

Constraints for complete work weekends are common in previous research, along with constraints restricting the number of work weekends. We do not include the latter in this formulation, as off-weekends are planned a long time in advance in Denmark. In one of the wards we have worked with, they create a preliminary schedule for off-weekends for an entire calendar year. Uprooting such traditions by giving a model for short term planning control over weekend work would likely lead to severe dissatisfaction from the employees, which are used to knowing their off-weekends a long time in advance.

In addition to working both Saturday and Sunday, the employees should generally work a fixed pattern of shifts, for example, the same shift on both days. Furthermore, an assignment on Friday may be a part of the weekend pattern, depending on the specific shift and Table 4 presents some examples of weekend patterns.

Table 4 Examples of weekend patterns. For day shifts, the weekend pattern only includes Saturday and Sunday, but for evening and night shifts we should also consider Friday as part of the pattern.

Friday	Saturday	Sunday
	Day	Day
Evening	Evening	Evening
Night	Night	Night

The constraint we use for scheduling combinations of assignments is extremely flexible. We can either enforce it with a bi-implication, i.e., scheduling neither assignment without the other, or with a single implication, i.e., perhaps scheduling one assignment without the other but not vice versa. An example of a single implication is when the coverage requirements differ between weekdays and weekends. Then we would enforce a single implication constraint, ensuring that employees working Saturday evening do also work Friday evening, while allowing some employees working on Friday evening to get the weekend off. The classic example for bi-implication would be an employee working the same shift on both Saturday and Sunday.

This section has described the use of this constraint for a specific employee. In addition, we also utilize it to schedule combinations for separate employees when matching shifts to compose a coverage, as presented in the next section.

3.2.6 Coverage constraints

The formulation includes several constraints related to coverage, i.e., assigning employees to shifts to satisfy predetermined staffing requirements. Each coverage constraint corresponds to a day and a shift (either shift type or shift set). Furthermore, a coverage constraint may be defined for specific skill categories and often these constraints are used not only to ensure sufficient staffing but also to manage the combination of skills available.

We enforce hard constraints for both a minimum and a maximum coverage. To ensure the feasibility of the minimum coverage, we can assign float nurses subject to a high penalty). For some shifts, the maximum coverage may be preferred and we assign a reward for achieving it. We note that this reward is substantially lower than the penalty for float nurses, i.e., we will not achieve the preferred coverage by adding float nurses.

As presented in Section 3.1.1, shifts may partly correspond to the coverage, i.e., as shift types *DE* and *EN* do for an *Evening* coverage constraint. By matching these shift types we achieve more flexible coverage constraints, and Figure 3 presents some different assignments that contribute toward an evening coverage. As employees assigned to either *DE* or *EN* only contribute towards a specific time interval, we include matching constraints (similar to those for weekend patterns) to ensure that we assign the same number of employees to both shift types.

In their categorization, De Causmaecker and Vanden Berghe (2011) included a category for coverage constraints defined over time intervals (e.g., a given hour of the day). Only a few researchers have defined coverage constraints in this manner (Burke et al., 2006). The increased flexibility from allowing to match shifts when composing the coverage is closely related to this category. Nonetheless, the time intervals in this case generally correspond to a shift set, and thus we do not claim that the formulation matches this category.

3.2.7 Balancing the work

We include two balancing constraints in the formulation, that contribute towards distributing the workload evenly between the employees and promote a feeling of fairness.

The first balancing constraint is related to the coverage constraints, as it aims at balancing any staffing beyond the minimum required evenly over the horizon. For example, we can generally exceed the minimum number for the *Day* shift set so instead of having ten excess employees working one day and zero the next, we balance this excess over the horizon.

The second balancing constraint relates to a fair distribution of unpopular shifts between the employees. For example, night work is not popular and many employees have agreements stating an absolute maximum of night shifts that they should work. Even though a roster would respect the maximum, it would not be perceived as fair if one employee works exactly the maximum number of night shifts while another works no night shifts, especially if these employees have a similar bound. Thus, we should spread these shifts evenly between the employees, but at the same time relative to their maximum.

3.2.8 Chaperoning

Chaperoning constraints connect the rosters of two or more employees. Although these constraints are common in practice, De Causmaecker and Vanden Berghe (2011)'s notation revealed that most previous research excludes them.

In this formulation, we include two types of chaperoning constraints, which are highly valuable in educational wards. We assume that a subset of the employees, referred to as *trainees*, are partaking in an educational program. Each trainee has a corresponding *chaperone* and we should regularly schedule a *chaperoning shift* for the two. During this shift, neither the trainee nor the chaperone are caring for patients, but instead, they are assessing the progress of the education along with planning the next steps. Additionally, we have constraints that promote that the trainees are assigned to patient care during the same time as their chaperones or other senior nurses, to further strengthen their education.

3.2.9 New constraints

The formulation includes three types of constraints that have never been included in previous research, clearly exhibiting the importance of collaborating closely with practitioners to match their needs. Two of these constraints promote healthy work schedules by allowing less frequent changes in sleeping patterns, while the third promotes days off around weekend work.

The first constraint restricts the number of different types of shifts we assign to an employee on consecutive days. For example, assigning an employee to day, evening and night shifts in the same week requires rapid changes in sleeping patterns. These types of irregularities have been reported to have a substantial impact on employees' health, not only short-term but also long-term. Thus, we penalize when assigning shifts from all these three shift sets too close together.

Table 5 presents an example where two employees are assigned to shifts from all three shift sets during one week. By employing simple swaps, we provide two alternative rosters that cover the same shifts. These alternatives assign at most two of the aforementioned shift sets to each employee.

The second constraint restricts the number of *work sequences* on consecutive days that include a given shift, where a work sequence refers to a series of consecutive workdays for an employee. These constraints are used to ensure that the employees work a compact series of

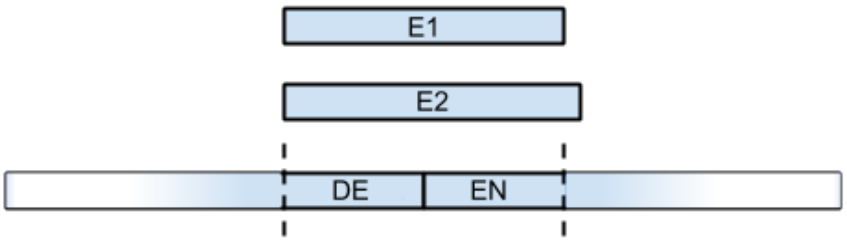

Fig. 3 Different combination of shift types corresponding to *Evening* coverage. By allowing for partial coverage, we obtain increased flexibility to satisfy the staffing requirements.

Table 5 Examples of distributing different shifts between employees.

Description	Employee	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
3 different shifts	e_1	Evening	Off	Off	Day	Night	Night	Night
	e_2	Day	Night	Off	Off	Evening	Evening	Evening
2 different shifts	e_1	**Day**	Off	Off	Day	Night	Night	Night
	e_2	**Evening**	Night	Off	Off	Evening	Evening	Evening
2 different shifts	e_1	Evening	**Night**	Off	**Off**	Night	Night	Night
	e_2	Day	**Off**	Off	**Day**	Evening	Evening	Evening

night shifts followed by a longer break from such shifts, rather than having isolated night shifts scattered.

Table 6 presents two examples of different distributions of night shifts on a weekly roster. The first example includes only two night shifts, which are either isolated on Monday and Thursday or placed consecutively on Wednesday and Thursday. The second example includes four night shifts and a single day shift, where the night shifts are either split into two sequences or performed consecutively.

When the night shifts are split into two sequences, the employees lack the opportunity to regulate their sleep in between their night shifts. Therefore, combining the night shifts into a single sequence contributes towards the health of the employees, which positively affects both short-term performance and long-term well-being.

The last constraint states that employees working some periods of the rostering horizon should get the surrounding days off, which is used to grant employees days off before and after weekend work. To clarify these constraints using an example, we assume that the employees should get a single day off both before and after the weekend. If we analyze the weekend patterns presented in Table 4 (Section 3.2.5) we see that the day off after the weekend is Monday, independent of the pattern. However, the day off before the weekend is either Thursday (for evening and night patterns) or Friday (for day patterns).

3.3 Objective function

The overall objective of the problem is to maximize the satisfaction with the roster, such that an experienced manager can look at and be pleased with the outcome. The objective function is a complex linear combination of penalties and rewards that minimizes the violation of soft constraints while maximizing the fulfillment of requests and preferences.

Each soft constraint has an associated weight, representing its relative importance. As an example, the weight associated with employing float nurses is significantly higher than the weight for deviating from an employee's weekly target. As the rostering data may differ substantially between horizons (for example due to specific requests), we do not use static weights. Instead, we manually tune them for each instance, to ensure that the resulting roster matches the requirements of the practitioners. These requirements relate to the combination of overall violations, along with a fair distribution between the employees.

4 Mathematial model

This section introduces the MIP formulation used to model the problem described in the previous section. Table 7 presents the notation used for sets and Table 8 the notation used for parameters.

We introduce the assignment variables $x_{e,d,s} \in \{0,1\}$, to denote whether we assign employee $e \in \mathcal{E}^{all}$ to shift type $s \in \mathcal{S}_{e,d}$ on day $d \in \mathcal{D}$. These are the main de-

Table 6 Examples of distributing night shifts.

Description	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
2 night shifts	**Night**	Off	Off	**Night**	Off	Off	Off
	Off	Off	**Night**	**Night**	Off	Off	Off
4 night shifts	**Night**	Off	Off	Day	**Night**	**Night**	**Night**
	Day	Off	Off	**Night**	**Night**	**Night**	**Night**

Table 7 Sets used in the MIP model

Set	Description
$\mathcal{E}$	Employees for whom we should create a full roster.
$\mathcal{E}^{all}$	All employees in the ward, including those that the model generally does not schedule (unless explictly requested).
$\mathcal{D}$	Days of the rostering horizon.
$\mathcal{D}^{pre}$	Days from the previous horizon.
$\mathcal{D}^{all}$	All days $\mathcal{D}^{all} = \mathcal{D} \cup \mathcal{D}^{pre}$ that we consider,
$\mathcal{W}$	Weeks of the rostering horizon.
$\mathcal{D}_w$	Days $\mathcal{D}_w \subseteq \mathcal{D}$ of week $w \in \mathcal{W}$.
$\mathcal{S}$	Shift types.
$\mathcal{S}^{work}$	Work shift types $\mathcal{S}^{work} \subset \mathcal{S}$.
$\mathcal{S}_{e,d}$	Shift types $\mathcal{S}_{e,d} \subseteq \mathcal{S}$ that are feasible for employee $e \in \mathcal{E}^{all}$ on day $d \in \mathcal{D}$, or the shift types that were assigned if $d \in \mathcal{D}^{pre}$.
$\mathcal{A}$	Feasible assignments for the rostering horizon $\mathcal{A} = \mathcal{E}^{all} \times \mathcal{D} \times \mathcal{S}_{e,d}$.
$\mathcal{A}^{all}$	Feasible assignments including those that were assigned during previous horizon $\mathcal{A}^{all} = \mathcal{E}^{all} \times \mathcal{D}^{all} \times \mathcal{S}_{e,d}$.
Γ	Conflict cliques.
$\mathcal{A}_\gamma$	Subset of assignments $\mathcal{A}_\gamma \subset \mathcal{A}$ belonging to conflict clique $\gamma \in \Gamma$.
$\Gamma_{e,d}$	Work conflict cliques for employee $e \in \mathcal{E}$ on day $d \in \mathcal{D}^{all}$
$\Gamma^{desc}_{e,d}$	A descending ordering of the cliques $\gamma \in \Gamma_{e,d}$ with respect to the number of conflicting assignments $\lvert\mathcal{A}_\gamma\rvert$.
$\mathcal{Z}$	Shift sets.
$\mathcal{Y}$	Patterns that we should forbid or penalize. Each pattern $y \in \mathcal{Y}$ represents a series of shift sets $\langle \sigma_1, \dots \sigma_{l_y} \rangle$ with $\sigma_i \in \mathcal{Z}$, $\forall i \in \{1, \dots, l_y\}$.
$\mathcal{C}$	Coverage constraints.
Ξ_j	Positions with competences for coverage $j \in \mathcal{C}$.
$\mathcal{Z}^{spread}$	Subset of shift sets $\mathcal{Z}^{spread} \subseteq \mathcal{Z}$, for which we should spread the assignments evenly among the employees.
$\mathcal{E}^{chap}$	Chaperones $\mathcal{E}^{chap} \subset \mathcal{E}^{all}$.
$\mathcal{E}^{train}_c$	Trainees $\mathcal{E}^{train}_c \subset \mathcal{E}^{all}$ with chaperone $c \in \mathcal{E}^{chap}$.
$\mathcal{E}^{G}$	Pairs of employees $\mathcal{E}^{G} \subset \mathcal{E} \times \mathcal{E}^{all}$ that we should assign work together.
$\mathcal{Z}^{together}$	Subset of shift sets $\mathcal{Z}^{together} \subseteq \mathcal{Z}$ where employees assigned to the same set are considered working together.
$\mathcal{Z}^{diff}$	Subset of shift sets $\mathcal{Z}^{diff} \subseteq \mathcal{Z}$ which we consider when restricting the assignments to different shift sets on consecutive days.
$\mathcal{Z}^{seq}_e$	Subset of shift sets $\mathcal{Z}^{seq}_e \subseteq \mathcal{Z}$ where we restrict the number of sequences on consecutive days containing certain shifts.
$\mathcal{Q}_e$	Periods where we prefer to assign surrounding days off if employee $e \in \mathcal{E}$ is working in the period.
$\mathcal{O}_{e,q}$	Assignments $\mathcal{O}_{e,q} \subset \mathcal{S}^{work} \times \mathcal{D}^{all}$ that overlap with period $q \in \mathcal{Q}_e$ for employee $e \in \mathcal{E}$.

cision variables of the problem. Additionally, we define numerous variables as listed in Table 9. Although many of these variables represent binary (or integer) occurrences, we can relax them to be continuous.

To ensure feasibility in continuation of the previous roster, we use an extension of the assignment variables in some constraints. We let $x^{pre}_{e,d,s} \in \{0,1\}$ denote the (fixed) solution from the previous horizon, i.e., whether employee $e \in \mathcal{E}^{all}$ was assigned to shift type $s \in \mathcal{S}$ on day $d \in \mathcal{D}^{pre}$. We let $\tilde{x}_{e,d,s}$ be the extension of $x_{e,d,s}$ to $\mathcal{E}^{all} \times \mathcal{D}^{all} \times \mathcal{S}_{e,d}$ by setting $\tilde{x}_{e,d,s} = x^{pre}_{e,d,s}$ for $e \in \mathcal{E}^{all}, d \in \mathcal{D}^{pre}$ and $s \in \mathcal{S}$. We note that this extension does not increase the number of variables or the complexity of the problem.

The remainder of this section presents the mathematical formulation as follows: Section 4.1 introduce cliques, which we employ to strengthen the formulation. Section 4.2 presents the constraints (following the same structure as Section 3.2) and Section 4.3 provides the objective. At last, Section 4.4 includes a discussion, drawing out specific aspects of this formulation and comparing them to the literature.

Table 8 Parameters used in the MIP model

Parameter	Description
$x^{pre}_{e,d,s}$	A binary parameter denoting whether employee $e \in \mathcal{E}^{all}$ had shift type $s \in \mathcal{S}$ on day $d \in \mathcal{D}^{pre}$ in the previous roster.
ρ^{full}	The minimum time for full rest between assignments (required to identify conflicting assignments).
ρ^{red}	The minimum time for reduced rest between assignments (required to identify conflicting assignments).
$T^{start}_{s,d}$	The starting time of shift type $s \in \mathcal{S}$ on day $d \in \mathcal{D}^{all}$.
$T^{end}_{s,d}$	The ending time of shift type $s \in \mathcal{S}$ on day $d \in \mathcal{D}^{all}$.
T^{prev}	A reference point in time from previous rostering horizon.
T^{fut}	A reference point in time from upcoming rostering horizon.
ρ^{pf}_i	The minimum number of hours off for $i \in \mathbb{N}$ consecutive protected days off.
v^{pref}	The maximum number of consecutive days without assigning any protected days off.
T^{target}_e	The number of hours we should assign to employee $e \in \mathcal{E}$ during the rostering horizon.
$T^{hour}_{e,d,s}$	The number of hours that assigning shift type $s \in \mathcal{S}$ on day $d \in \mathcal{D}$ counts towards the target for employee $e \in \mathcal{E}$.
N^+	The maximum positive deviation from the target hours before penalizing heavier in two-factor penalty.
N^-	The maximum negative deviation from the target hours before penalizing heavier in two-factor penalty.
$T^{week}_{e,w}$	The number of hours we should assign to employee $e \in \mathcal{E}$ during week $w \in \mathcal{W}$.
$\alpha_{s,\sigma}$	The relation between shift set $\sigma \in \mathcal{Z}$ and shift type $s \in \mathcal{S} \cap \sigma$.
$M^{total,ub}_{e,\sigma}$	The maximum number of assignments employee $e \in \mathcal{E}$ should get from shift set $\sigma \in \mathcal{Z}$ during the rostering horizon.
$M^{total,fix}_{e,\sigma}$	The fixed number of assignments employee $e \in \mathcal{E}$ should get from shift set $\sigma \in \mathcal{Z}$ during the rostering horizon.
$M^{week,fix}_{e,\sigma,w}$	The fixed number of assignments to shift set $\sigma \in \mathcal{Z}$ employee $e \in \mathcal{E}$ should get in week $w \in \mathcal{W}$.
m^{seq}	The minimum length of a work sequence in days.
$M^{row}_{e,d,\sigma}$	The maximum number of days in a row from day $d \in \mathcal{D}^{all}$ we should assign employee $e \in \mathcal{E}$ to shift set $\sigma \in \mathcal{Z}$.
$\beta^{row}_{e,d,\sigma}$	A binary parameter denoting whether the maximum assignments in a row from day $d \in \mathcal{D}$ for employee $e \in \mathcal{E}$ to shift set $\sigma \in \mathcal{Z}$ should be a hard ($\beta^{row}_{e,d,\sigma} = 1$) or a soft ($\beta^{row}_{e,d,\sigma} = 0$) constraint.
$H^{consec}_{e,d,\sigma}$	The maximum number of hours we can assign employee $e \in \mathcal{E}$ to shift set $\sigma \in \mathcal{Z}$ on $D^{consec}_{e,d,\sigma}$ consecutive days from day $d \in \mathcal{D}$.
$D^{consec}_{e,d,\sigma}$	The number of consecutive days when restricting the number of hours assigned to employee $e \in \mathcal{E}$ from shift set $\sigma \in \mathcal{Z}$ from day $d \in \mathcal{D}$.
l_y	The length of pattern $y \in \mathcal{Y}$ in days.
β^{pat}_y	A binary parameter denoting whether pattern $y \in \mathcal{Y}$ should be a hard ($\beta^{pat}_y = 1$) or a soft ($\beta^{pat}_y = 0$) constraint.
$\Pi_{e,d,y}$	A binary parameter denoting whether pattern $y \in \mathcal{Y}$ should be active for employee $e \in \mathcal{E}$ from day $d \in \mathcal{D}$.
$L^{emp,both}_{e,d_1,d_2,\sigma_1,\sigma_2}$	A binary parameter denoting whether we can only assign employee $e \in \mathcal{E}$ to shift set $\sigma_1 \in \mathcal{Z}$ on day $d_1 \in \mathcal{D}$ in combination with assigning him to $\sigma_2 \in \mathcal{Z}$ on day $d_2 \in \mathcal{D}$, and vice versa.
$L^{emp,one}_{e,d_1,d_2,\sigma_1,\sigma_2}$	A binary parameter denoting whether we can only assign employee $e \in \mathcal{E}$ to shift set $\sigma_1 \in \mathcal{Z}$ on day $d_1 \in \mathcal{D}$ in combination with assigning him to $\sigma_2 \in \mathcal{Z}$ on day $d_2 \in \mathcal{D}$, but not necessarily the other way around.
$c^{min}_{j,d}$	The minimum number of employees required for coverage $j \in \mathcal{C}$ on day $d \in \mathcal{D}$.
$c^{max}_{j,d}$	The maximum number of employees allowed for coverage $j \in \mathcal{C}$ on day $d \in \mathcal{D}$.
β^{float}_j	A binary parameter denoting whether a float nurse is allowed for coverage $j \in \mathcal{C}$.
ξ_e	The position of employee $e \in \mathcal{E}^{all}$.
$L^{ward,both}_{d_1,d_2,\sigma_1,\sigma_2}$	A binary parameter denoting whether we can only assign any employee to shift set $\sigma_1 \in \mathcal{Z}$ on day $d_1 \in \mathcal{D}$ in combination with assigning any employee to $\sigma_2 \in \mathcal{Z}$ on day $d_2 \in \mathcal{D}$, and vice versa.
$L^{ward,one}_{d_1,d_2,\sigma_1,\sigma_2}$	A binary parameter denoting whether we can only assign any employee to shift set $\sigma_1 \in \mathcal{Z}$ on day $d_1 \in \mathcal{D}$ in combination with assigning any employee to $\sigma_2 \in \mathcal{Z}$ on day $d_2 \in \mathcal{D}$, but not necessarily the other way around.
l^{gen}	The standard length of a shift type in hours.
$\tilde{M}^{total,ub}_{e,\sigma}$	An extension of $M^{total,ub}_{e,\sigma}$, defined for all $e \in \mathcal{E}$ and $\sigma \in \mathcal{Z}$. If $M^{total,ub}_{e,\sigma}$ is defined, then $\tilde{M}^{total,ub}_{e,\sigma} = M^{total,ub}_{e,\sigma}$. Otherwise $\tilde{M}^{total,ub}_{e,\sigma} = \left\lfloor \frac{T^{target}_e}{l^{gen}} \right\rfloor$.
n^{diff}	The maximum number of different shift sets from $\mathcal{Z}^{diff}$ we should assign on D^{diff} consecutive days.
D^{diff}	The number of consecutive days when restricting the number of different shift sets.
$\chi_{e,\sigma}$	The maximum number of work sequences including shift set $\sigma \in \mathcal{Z}^{seq}_e$ we should assign to employee $e \in \mathcal{E}$ on $D^{seq}_{e,\sigma}$ consecutive days.
$D^{seq}_{e,\sigma}$	The number of consecutive days when restricting the number sequences including the same shift set.
q^s	The first day of period $q \in \mathcal{Q}_e$ for employee $e \in \mathcal{E}$.
q^e	The last day of period $q \in \mathcal{Q}_e$ for employee $e \in \mathcal{E}$.
q^b	The number of days employee $e \in \mathcal{E}$ should have off before working in period $q \in \mathcal{Q}_e$.
q^a	The number of days employee $e \in \mathcal{E}$ should have off after working in period $q \in \mathcal{Q}_e$.
ψ_{d_1,d_2,s_1,s_2}	The number of full days between shift type $s_1 \in \mathcal{S}^{work}$ on day $d_1 \in \mathcal{D}$ and shift type $s_2 \in \mathcal{S}^{work}$ on day $d_2 \in \mathcal{D}$, where $d_1 \leq d_2$.

Table 9 Variables used in the MIP model

Variable	Description
$x_{e,d,s} \in \{0,1\}$	Denotes whether we assign employee $e \in \mathcal{E}^{all}$ to shift type $s \in \mathcal{S}_{e,d}$ on day $d \in \mathcal{D}$, i.e. whether we schedule assignment $(e,d,s) \in \mathcal{A}$.
$\tilde{x}_{e,d,s} \in \{0,1\}$	Extension of $x_{e,d,s}$ that includes the (fixed) assignments $x^{pre}_{e,d,s}$ from previous horizon.
$p_{e,d,i} \in \{0,1\}$	Denotes whether we assign employee $e \in \mathcal{E}$ to $i \in \mathbb{N}$ consecutive protected days off starting at day $d \in \mathcal{D}$.
$\tau^{last}_{e,d} \geq 0$	The number of hours from the reference point T^{prev} to the end of the last work shift employee $e \in \mathcal{E}$ has before day $d \in \mathcal{D}$.
$\tau^{next}_{e,d} \geq 0$	The number of hours from the start of the first work shift employee $e \in \mathcal{E}$ has after day $d \in \mathcal{D}$ to the reference point T^{fut}.
$v_{e,d} \in [0,1]$	Denotes whether employee $e \in \mathcal{E}$ has no protected day off from day $d \in \mathcal{D}^{all}$ to day $d + v^{pref} \in \mathcal{D}$, including both days.
$h_{e,w} \geq 0$	The number of hours we assign to employee $e \in \mathcal{E}$ during week $w \in \mathcal{W}$.
$t^{+}_{e} \in [0, N^{+}]$	The number of hours we assign to employee $e \in \mathcal{E}$ above the target hours, within a bound N^{+}.
$t^{++}_{e} \geq 0$	The number of hours we assign to employee $e \in \mathcal{E}$ above the target hours, exceeding N^{+}.
$t^{-}_{e} \in [0, N^{-}]$	The number of hours we assign to employee $e \in \mathcal{E}$ below the target hours, within a bound N^{-}.
$t^{--}_{e} \geq 0$	The number of hours we assign to employee $e \in \mathcal{E}$ below the target hours, exceeding N^{-}.
$\lambda^{+}_{e,w} \geq 0$	The number of hours we assign to employee $e \in \mathcal{E}$ during week $w \in \mathcal{W}$ above the weekly target.
$\lambda^{-}_{e,w} \geq 0$	The number of hours we assign to employee $e \in \mathcal{E}$ during week $w \in \mathcal{W}$ below the weekly target.
$\theta^{start}_{e,d} \in [0,1]$	Denotes whether employee $e \in \mathcal{E}$ has a work sequence starting on day $d \in \mathcal{D}^{all}$.
$\theta^{end}_{e,d} \in [0,1]$	Denotes whether employee $e \in \mathcal{E}$ has a work sequence ending on day $d \in \mathcal{D}^{all}$.
$\delta_{e,d} \geq 0$	The square of the violation of minimum work sequence when employee $e \in \mathcal{E}$ has a sequence below the minimum that ends on day $d \in \mathcal{D}$.
$\mu_{e,d,\sigma} \geq 0$	If $\beta^{row}_{e,d,\sigma} = 0$, this denotes whether we assign employee $e \in \mathcal{E}$ to shift set $\sigma \in \mathcal{Z}$ on more than $M^{row}_{e,d,\sigma}$ days in a row, starting on day $d \in \mathcal{D}^{all}$. If $\beta^{row}_{e,d,\sigma} = 1$, we fix this variable as zero.
$\pi_{e,d,y} \in \{0,1\}$	If $\beta^{pat}_{y} = 0$, this denotes whether we violate pattern $y \in \mathcal{Y}$ starting on day $d \in \mathcal{D}^{all}$ for employee $e \in \mathcal{E}$. If $\beta^{pat}_{y} = 1$, we fix this variable as zero.
$f_{j,d} \in \mathbb{N}_0$	If $\beta^{float}_{j} = 1$, this denotes the number of float nurses assigned to coverage $j \in \mathcal{C}$ on day $d \in \mathcal{D}$. If $\beta^{float}_{j} = 0$, we fix this variable as zero.
$c^{+}_{j} \geq 0$	The highest number of employees exceeding $c^{min}_{j,d}$ that we assign to coverage $j \in \mathcal{C}$ on any day $d \in \mathcal{D}$. If $j \in \mathcal{C}$ either has a maximum number of employees $c^{max}_{j,d}$ defined for all days $d \in \mathcal{D}$ or has a restriction regarding skills (i.e., $\Xi_j \neq \bigcup_{e \in \mathcal{E}^{all}} \xi_e$), then we fix the variable to be zero.
$c^{-}_{j} \geq 0$	The lowest number of employees exceeding $c^{min}_{j,d}$ that we assign to coverage $i \in \mathcal{C}$ on any day $d \in \mathcal{D}$. If $j \in \mathcal{C}$ either has a maximum number of employees $c^{max}_{j,d}$ defined for all days $d \in \mathcal{D}$ or has a restriction regarding skills (i.e., $\Xi_j \neq \bigcup_{e \in \mathcal{E}^{all}} \xi_e$), then we fix the variable to be zero.
$\zeta^{max}_{\sigma} \geq 0$	The maximum assignments to shift set $\sigma \in \mathcal{Z}^{spread}$ for any employee, relative to employee specific restrictions.
$\zeta^{min}_{\sigma} \geq 0$	The minimum assignments to shift set $\sigma \in \mathcal{Z}^{spread}$ for any employee, relative to employee specific restrictions.
$g_{e_1,e_2,d,\sigma} \in [0,1]$	Denotes how much employees $(e_1, e_2) \in \mathcal{E}^{G}$ assigned to shift set $\sigma \in \mathcal{Z}^{together}$ on day $d \in \mathcal{D}$ are working together.
$\kappa_{e,d,\sigma} \in [0,1]$	Denotes whether we assign employee $e \in \mathcal{E}$ to shift set $\sigma \in \mathcal{Z}^{diff}$ on any day $d' \in \{d, \dots d + D^{diff} - 1\}$ for $d \in \mathcal{D}^{all}$.
$\kappa^{more}_{e,d} \geq 0$	The number of shift sets from $\mathcal{Z}^{diff}$ we assign to employee $e \in \mathcal{E}$ on any day $d' \in \{d, \dots d + D^{diff} - 1\}$ for $d \in \mathcal{D}^{all}$ exceeding the maximum n^{diff}.
$\phi^{start}_{e,d,\sigma} \in [0,1]$	Denotes whether employee $e \in \mathcal{E}$ starts a work sequence including shift set $\sigma \in \mathcal{Z}^{seq}_{e}$ on day $d \in \mathcal{D}^{all}$.
$\phi^{end}_{e,d,\sigma} \in [0,1]$	Denotes whether employee $e \in \mathcal{E}$ ends a work sequence including shift set $\sigma \in \mathcal{Z}^{seq}_{e}$ on day $d \in \mathcal{D}^{all}$.
$\phi^{count}_{e,d,\sigma} \geq 0$	The number of work sequences exceeding $\chi_{e,\sigma}$ that we assign to employee $e \in \mathcal{E}$ on $D^{seq}_{e,\sigma}$ consecutive days from $d \in \mathcal{D}^{all}$, that include $\sigma \in \mathcal{Z}^{seq}_{e}$.
$B_{e,q} \geq 0$	The number of days off we assign to employee $e \in \mathcal{E}$ before period $q \in \mathcal{Q}$ if the employee is assigned work during the period.
$A_{e,q} \geq 0$	The number of days off we assign to employee $e \in \mathcal{E}$ after period $q \in \mathcal{Q}$ if the employee is assigned work during the period.

4.1 Cliques

When using exact approaches to solve MIP formulations, the size of the model and its tightness may have a substantial impact on the solution times. Therefore, we employ cliques to provide a tighter LP relaxation of some constraints while simultaneously reducing the number of constraints.

To formulate constraints for physically infeasible or illegal combinations of assignments, we define a set of *conflict cliques* Γ. To generate these cliques we employ the theory of *maximal cliques* from graph theory. We construct a *conflict graph* where each node corresponds to an assignment, i.e., a triple (e, d, s) of an employee, a day and a shift type. We let $\mathcal{A}$ denote the set of feasible assignments for the rostering horizon and let $\mathcal{A}^{all}$ be the natural extension to previous rostering horizon.

We define a pair of conflicting assignments as two assignments $(e_1, d_1, s_1), (e_2, d_2, s_2) \in \mathcal{A}^{all}$ where assigning both of them would lead to an infeasible roster. For each pair we add an edge to connect the corresponding nodes. A clique $\gamma \in \Gamma$ corresponds to a subset of assignments $\mathcal{A}_\gamma \subset \mathcal{A}^{all}$, such that every two assignments in $\mathcal{A}_\gamma$ are conflicting.

We choose Γ to be a clique cover of the graph, meaning that for every conflicting pair of assignments at least one clique contains both assignments. We automatically generate these cliques using the heuristic by Kou et al. (1978), which produces a clique cover while trying to minimize the number of cliques $|\Gamma|$.

In addition to Γ, we also generate sets of work conflict cliques $\Gamma_{e,d}$ for each employee e and day d of the rostering horizon. We employ these cliques in some constraints related to consecutive days.

We generate these cliques using Algorithm 1, where we first iterate over all $\gamma \in \Gamma$ to create cliques $\Gamma_{e,d}$ with all work assignments in $\mathcal{A}_\gamma$ that correspond to $e \in \mathcal{E}$ and $d \in \mathcal{D}$. Afterwards, we try to reduce $|\Gamma_{e,d}|$ by iterating through $\Gamma^{desc}_{e,d}$, defined as a descending ordering of the cliques $\gamma \in \Gamma_{e,d}$ w.r.t. the number of conflicting assignments $|\mathcal{A}_\gamma|$. For $d \in \mathcal{D}^{pre}$, we define one $\gamma \in \Gamma_{e,d}$ such that $\mathcal{A}_\gamma$ only includes the assignment that employee $e \in \mathcal{E}$ got on day $d \in \mathcal{D}^{pre}$.

To further clarify the generation of work conflict cliques we provide a simple example in Table 10, where we generate work conflict cliques for day $d \in \mathcal{D}$. We let s_D, s_E and s_N denote day, evening and night shifts, respectively. Furthermore, s_c (the chaperoning shift) is categorized as a work shift, while s_{pf} and s_{off} are not. At last, $d-1$ indicates that the shift is positioned on the day before, and $d+1$ on the day after, as a clique $\gamma \in \Gamma$ does not have to be confined to a single day.

Algorithm 1 Generation of work conflict cliques

1: $\Gamma_{e,d} = \emptyset \quad \forall e \in \mathcal{E}, d \in \mathcal{D}$
2: **for** $\gamma \in \Gamma$ **do**
3: **for** $e \in \mathcal{E}, d \in \mathcal{D}$ **do**
4: $\Gamma_{e,d} \leftarrow \Gamma_{e,d} \bigcup \{(e', d', s) \in \mathcal{A}_\gamma \,|\, e' = e \wedge d' = d \wedge s \in \mathcal{S}^{work}\}$
5: **end for**
6: **end for**
7: **for** $(e_1, d_1, s_1), (e_2, d_2, s_2) \in \mathcal{A}$ **do**
8: **if** $\exists \gamma \in \Gamma : \{(e_1, d_1, s_1), (e_2, d_2, s_2)\} \subseteq \mathcal{A}_\gamma$ **then**
9: $((e_1, d_1, s_1), (e_2, d_2, s_2))$ marked as *uncovered*
10: **end if**
11: **end for**
12: **for** $e \in \mathcal{E}, d \in \mathcal{D}$ **do**
13: **for** $\gamma \in \Gamma^{desc}_{e,d}$ **do**
14: **if** $\exists (e_1, d_1, s_1), (e_2, d_2, s_2) \in \mathcal{A}_\gamma$: $((e_1, d_1, s_1), (e_2, d_2, s_2))$ marked *uncovered* **then**
15: **for** $(e_1, d_1, s_1), (e_2, d_2, s_2) \in \mathcal{A}_\gamma$ **do**
16: $((e_1, d_1, s_1), (e_2, d_2, s_2))$ marked as *covered*
17: **end for**
18: **else**
19: $\Gamma_{e,d} \leftarrow \Gamma_{e,d} \backslash \{\gamma\}$
20: **end if**
21: **end for**
22: **end for**

Table 10 presents the first step in generating the work conflict cliques, namely identifying the relevant cliques (γ_1-γ_4) and extracting the work assignments. In each column, the highlighted shift types make up an item in $\Gamma_{e,d}$ after line 6 (Algorithm 1).

Table 10 The first step in generating work conflict cliques.

γ_1	γ_2	γ_3	γ_4
s_E $(d-1)$	s_N $(d-1)$	s_{off}	**s_N**
s_D	s_{pf}	s_{pf}	s_{pf} $(d+1)$
s_c	**s_D**	**s_D**	
	s_c	**s_c**	
	s_E	**s_E**	
		s_N	

We sort these cliques in a descending order, namely $\gamma_3, \gamma_2, \gamma_1, \gamma_4$. After initially marking all conflicting pairs of assignments as uncovered (lines 7-11), we go through the cliques in the descending order. In γ_3 we have four shift types, corresponding to six conflicting pairs that we mark as covered. When moving through the remaining cliques, all pairs have been marked as covered and thus we can remove those cliques from the set of cliques. As a result, only one clique remains in the set of work conflict cliques, namely $\Gamma_{e,d} = \{\gamma_3\}$.

4.2 Constraints

We now present the constraints of the model. The first constraint is to assign at least one shift a day, as given with Equation (1).

$$\sum_{s \in \mathcal{S}_{e,d}} x_{e,d,s} \geq 1, \quad \forall e \in \mathcal{E}, d \in \mathcal{D} \tag{1}$$

4.2.1 Physically infeasible or illegal combinations

We prevent conflicting pairs of assignments with Equation (2), which employs conflict cliques. This constraint, along with the automatic generation of cliques, provides a more flexible alternative to common constraints from previous research.

$$\sum_{(e,d,s) \in \mathcal{A}_\gamma} \tilde{x}_{e,d,s} \leq 1, \quad \forall \gamma \in \mathit{\Gamma} \tag{2}$$

4.2.2 Ensuring days off

The following constraints ensure that we meet the rules regarding protected days off. As we have defined days off as shift types, all other constraints required for days off are presented in a general matter (for example in Section 4.2.3).

We let $s_{pf} \in \mathcal{S}$ denote the shift type for a protected day off and note that Equation (2) ensures that if we assign $s_{pf} \in \mathcal{S}$, then it is the only shift assigned to that employee on that day.

To keep track of the protected days off, we introduce decision variables $p_{e,d,i} \in \{0,1\}$ denoting whether employee $e \in \mathcal{E}$ starts a sequence of $i \in \mathbb{N}$ protected days off on day $d \in \mathcal{D}^{all}$. We link these variables to the assignment variables with Equations (3)-(5).

$$\sum_{\substack{d' \in \mathcal{D}^{all}, \\ i \in \mathbb{N}: \\ d-i < d' \leq d}} p_{e,d',i} = \tilde{x}_{e,d,s_{pf}}, \quad \forall e \in \mathcal{E}, d \in \mathcal{D}^{all} : s_{pf} \in \mathcal{S}_{e,d} \tag{3}$$

$$\sum_{\substack{d' \in \mathcal{D}^{all}, \\ i \in \mathbb{N}: \\ d-i < d' \leq d}} p_{e,d',i} = 0, \quad \forall e \in \mathcal{E}, d \in \mathcal{D}^{all} : s_{pf} \notin \mathcal{S}_{e,d} \tag{4}$$

$$\sum_{j \in \mathbb{N}} p_{e,d,j} + \sum_{\substack{d' \in \mathcal{D}^{all}, \\ i \in \mathbb{N}: \\ d'+i=d}} p_{e,d',i} \leq 1, \quad \forall e \in \mathcal{E}, d \in \mathcal{D}^{all} \tag{5}$$

Equations (3)-(4) ensure that the $p_{e,d,i}$ variable can only be one if we have assigned employee $e \in \mathcal{E}$ to the protected day off shift $s_{pf} \in \mathcal{S}$ on all days from $d \in \mathcal{D}^{all}$ to $d+i \in \mathcal{D}$, including both days. Furthermore, Equations (5) ensure that two sequences of protected days off are never adjacent, but instead defined as one longer sequence.

Equations (6)-(10) ensure a sufficient number of uninterrupted hours off for protected days off by keeping track of the end time of a previous shift and the starting time for an upcoming shift. We note that T^{prev}, and T^{fut}, denote reference points in time from previous, and upcoming, rostering horizons. As these are used as a baseline, their exact values are irrelevant. When comparing to these reference values, we round on hours to two decimal places, denoted with $\lfloor \cdot \rceil_2$ in the formulation.

$$\tau^{last}_{e,d} - \sum_{\substack{s \in \mathcal{S}^{work}: \\ (e,d-1,s) \in \mathcal{A}_\gamma}} \left\lfloor T^{end}_{s,d-1} - T^{prev} \right\rceil_2 \cdot \tilde{x}_{e,d-1,s} \geq 0, \quad \forall e \in \mathcal{E}, d \in \mathcal{D}^{all}, \gamma \in \mathit{\Gamma}_{e,d} \tag{6}$$

$$\tau^{last}_{e,d} \geq \tau^{last}_{e,d-1}, \quad \forall e \in \mathcal{E}, d \in \mathcal{D}^{all} \tag{7}$$

$$\tau^{next}_{e,d} - \sum_{\substack{s \in \mathcal{S}^{work}: \\ (e,d+1,s) \in \mathcal{A}_\gamma}} \left\lfloor T^{start}_{s,d+1} - T^{fut} \right\rceil_2 \cdot \tilde{x}_{e,d+1,s} \leq \left\lfloor T^{fut} - T^{prev} \right\rceil_2, \quad \forall e \in \mathcal{E}, d \in \mathcal{D}^{all}, \gamma \in \mathit{\Gamma}_{e,d} \tag{8}$$

$$\tau^{next}_{e,d} \leq \tau^{next}_{e,d+1}, \quad \forall e \in \mathcal{E}, d \in \mathcal{D}^{all}, d+1 \in \mathcal{D}^{all} \tag{9}$$

$$\tau^{last}_{e,d} + \sum_{i \in \mathbb{N}} \rho^{pf}_i \cdot p_{e,d,i} \leq \tau^{next}_{e,d}, \quad \forall e \in \mathcal{E}, d \in \mathcal{D}^{all} \tag{10}$$

When we assign employee $e \in \mathcal{E}$ to a work shift $s \in \mathcal{S}^{work}$ on day $d-1 \in \mathcal{D}^{all}$, then Equations (6) ensure that the counter variable $\tau^{last}_{e,d} \geq 0$ is updated according to the end of the work shift. Similarly, Equations (8) ensure that the counter variable $\tau^{next}_{e,d} \geq 0$ is updated according to the start of a work shift on day $d+1 \in \mathcal{D}^{all}$. Equations (7) and (9) ensure the consistency of the variables on days where we do not assign a work shift on the previous or subsequent day. Finally, Equations (10) ensure that the variable $p_{e,d,i}$ can only become one when the difference between the counters $\tau^{next}_{e,d}$ and $\tau^{prev}_{e,d}$ is at least the required minimum hours for $i \in \mathbb{N}$ protected days off.

The objective function penalizes when we assign protected days of too far apart. Equation (11) connects the penalized variable $v_{e,d} \in [0,1]$ to the distance between protected days off.

$$v_{e,d} + \sum_{\substack{d' \in \mathcal{D}: \\ d \le d' < d + v^{pref}}} \tilde{x}_{e,d',s_{pf}} \ge 1, \quad \forall e \in \mathcal{E}, d \in \mathcal{D}^{all} : d + v^{pref} \in \mathcal{D} \tag{11}$$

4.2.3 Restricting the overall work

Equation (12) keeps track of the hours assigned each week and Equations (13)-(14) compare it to the contractual hours and the weekly targets, respectively. The deviations measured in Equations (13)-(14) are penalized in the objective function. We note that Equation 13 uses a two-factor penalty such that extreme deviations are penalized extensively compared to small deviations.

$$\sum_{\substack{d \in \mathcal{D}_w, \\ s \in \mathcal{S}_{e,d}}} T^{hour}_{e,d,s} \cdot x_{e,d,s} = h_{e,w}, \quad \forall e \in \mathcal{E}, w \in \mathcal{W} \tag{12}$$

$$\sum_{w \in \mathcal{W}} h_{e,w} - t^{+}_{e} + t^{-}_{e} - t^{++}_{e} + t^{--}_{e} = T^{target}_{e}, \quad \forall e \in \mathcal{E} \tag{13}$$

$$h_{e,w} - \lambda^{+}_{e,w} + \lambda^{-}_{e,w} = T^{week}_{e,w}, \quad \forall e \in \mathcal{E}, w \in \mathcal{W} \tag{14}$$

Equation (15) provides an upper bound for assigning specific shifts, while Equation (16) ensures a fixed number of assignments to a given shift during the rostering horizon. Furthermore, Equation (17) ensures a fixed number of assignment to a given shift during a specific week of the rostering horizon.

$$\sum_{\substack{d \in \mathcal{D}, \\ s \in \mathcal{S}_{e,d} \cap \sigma}} \alpha_{s,\sigma} \cdot x_{e,d,s} \le M^{total,ub}_{e,\sigma}, \quad \forall e \in \mathcal{E}, \sigma \in \mathcal{Z} : M^{total,ub}_{e,\sigma} \text{ defined} \tag{15}$$

$$\sum_{\substack{d \in \mathcal{D}, \\ s \in \mathcal{S}_{e,d} \cap \sigma}} \alpha_{s,\sigma} \cdot x_{e,d,s} = M^{total,fix}_{e,\sigma}, \quad \forall e \in \mathcal{E}, \sigma \in \mathcal{Z} : M^{total,fix}_{e,\sigma} \text{ defined} \tag{16}$$

$$\sum_{\substack{d \in \mathcal{D}_w, \\ s \in \mathcal{S}_{e,d} \cap \sigma}} \alpha_{s,\sigma} \cdot x_{e,d,s} = M^{week,fix}_{e,\sigma,w}, \quad \forall e \in \mathcal{E}, \sigma \in \mathcal{Z}, w \in \mathcal{W} : M^{week,fix}_{e,\sigma,w} \text{ defined} \tag{17}$$

4.2.4 Consecutive work assignments

Equations (18)-(20) measure whether a work sequence is shorter than the preferred minimum.

$$\sum_{\substack{s \in \mathcal{S}^{work}: \\ (e,d,s) \in \mathcal{A}_\gamma}} \tilde{x}_{e,d,s} - \sum_{s \in \mathcal{S}^{work}} \tilde{x}_{e,d-1,s} \le \theta^{start}_{e,d}, \quad \forall e \in \mathcal{E}, d \in \mathcal{D}^{all}, \gamma \in \Gamma_{e,d} \tag{18}$$

$$\sum_{\substack{s \in \mathcal{S}^{work}: \\ (e,d,s) \in \mathcal{A}_\gamma}} \tilde{x}_{e,d,s} - \sum_{s \in \mathcal{S}^{work}} \tilde{x}_{e,d+1,s} \le \theta^{end}_{e,d}, \quad \forall e \in \mathcal{E}, d \in \mathcal{D}^{all}, \gamma \in \Gamma_{e,d} : d+1 \in \mathcal{D} \tag{19}$$

$$\left(m^{seq} - d + d' - 1\right)^2 \cdot \left(\theta^{end}_{e,d} + \theta^{start}_{e,d'} - 1\right) \le \delta_{e,d}, \quad \forall e \in \mathcal{E}, d \in \mathcal{D}, d' \in \mathcal{D}^{all} : d - m^{seq} < d' \le d \tag{20}$$

Equations (18) force the variables $\theta^{start}_{e,d} \in [0,1]$ to be one if employee $e \in \mathcal{E}$ has a work shift on day $d \in \mathcal{D}^{all}$ but not on day $d-1 \in \mathcal{D}^{all}$. Similarly, Equations (19) force $\theta^{end}_{e,d} \in [0,1]$ to be one if employee $e \in \mathcal{E}$ has a work shift on day $d \in \mathcal{D}^{all}$ but not on day $d+1 \in \mathcal{D}^{all}$. Finally, Equations (20) consider work sequences shorter than preferred and bind the penalty variable $\delta_{e,d} \ge 0$ to be the square of the deviation, to ensure that extremely short sequences, e.g., single work days, are heavily penalized.

Equation (21) considers the maximum consecutive assignment to a given shift set. These constraints can either be enforced as hard constraints by fixing the variable $\mu_{e,d,\sigma} \ge 0$ to zero, or as soft constraints by penalizing the variable in the objective. Equation (22) restricts the number of hours we may assign to an employee during a set of consecutive days.

$$\sum_{\substack{d' \in \mathcal{D}^{all}, \\ s \in \mathcal{S}_{e,d} \cap \sigma: \\ d \le d' \le d + M^{row}_{e,d,\sigma}}} \alpha_{s,\sigma} \cdot \tilde{x}_{e,d',s} - \mu_{e,d,\sigma} \le M^{row}_{e,d,\sigma}, \quad \forall e \in \mathcal{E}, d \in \mathcal{D}^{all}, \sigma \in \mathcal{Z} : M^{row}_{e,d,\sigma} \text{ defined} \tag{21}$$

$$\sum_{\substack{d \le d' \le d + D^{consec}_{e,d,\sigma} - 1, \\ s \in \sigma \cap \mathcal{S}_{e,d'}}} T^{hour}_{e,d',s} \cdot \tilde{x}_{e,d',s} \le H^{consec}_{e,d,\sigma}, \quad \forall e \in \mathcal{E}, d \in \mathcal{D}^{all}, \sigma \in \mathcal{Z} : H^{consec}_{e,d,\sigma} \text{ and } D^{consec}_{e,d,\sigma} \text{ defined.} \tag{22}$$

4.2.5 Patterns

Equation (23) prevents or penalizes unwanted series of assignments in the roster. We note that these con-

straints require binary variables $\pi_{e,d,y} \in \{0,1\}$, as opposed to continuous variables as most constraints. Therefore, overusing this type of constraint may negatively impact the complexity of the problem, making it harder for the exact solver.

$$\sum_{\substack{i\in\{1,\dots,l_y\},\\ s_i\in\sigma_i\cap\mathcal{S}_{e,d+i-1}}} \tilde{x}_{e,d+i-1,s_i} \leq l_y - 1 + \pi_{e,d,y},$$
$$\forall e \in \mathcal{E}, d \in \mathcal{D}^{all}, y \in \mathcal{Y}:$$
$$d + l_y - 1 \in \mathcal{D}, \Pi_{e,d,y} = 1 \tag{23}$$

Equations (24)-(25) ensure that some assignments are only scheduled in a combination with other assignments, where Equation (24) enforces bi-implication and Equation (25) enforces a single implication.

$$\sum_{s\in\mathcal{S}_{e,d_1}\cap\sigma_1} \tilde{x}_{e,d_1,s} = \sum_{s\in\mathcal{S}_{e,d_2}\cap\sigma_2} \tilde{x}_{e,d_2,s},$$
$$\forall e \in \mathcal{E}, d_1, d_2 \in \mathcal{D}^{all}, \sigma_1, \sigma_2 \in \mathcal{Z}:$$
$$L^{emp,both}_{e,d_1,d_2,\sigma_1,\sigma_2} = 1 \tag{24}$$
$$\sum_{s\in\mathcal{S}_{e,d_1}\cap\sigma_1} \tilde{x}_{e,d_1,s} \leq \sum_{s\in\mathcal{S}_{e,d_2}\cap\sigma_2} \tilde{x}_{e,d_2,s},$$
$$\forall e \in \mathcal{E}, d_1, d_2 \in \mathcal{D}^{all}, \sigma_1, \sigma_2 \in \mathcal{Z}:$$
$$L^{emp,one}_{e,d_1,d_2,\sigma_1,\sigma_2} = 1 \tag{25}$$

4.2.6 Coverage constraints

We let $\mathcal{C}$ denote the set of coverage constraints and without loss of generality we assume that every coverage $j \in \mathcal{C}$ has an associated shift set $\sigma_j \in \mathcal{Z}$.

Equation (26) expresses a minimum staffing requirement and Equation (27) ensures that we do not exceed the allowed maximum staffing, where the variables $f_{j,d} \in \mathbb{N}_0$ represent float nurses. While the use of float nurses may be necessary, the objective function heavily penalizes this variable. Furthermore, Equations (28)-(29) ensure correct matching in case of partial coverage.

$$\sum_{\substack{e\in\mathcal{E}^{all},\\ s\in\sigma_j:\\ \xi_e\in\Xi_j}} \alpha_{s,\sigma_j}\cdot x_{e,d,s} + f_{j,d} \geq c^{min}_{j,d},$$
$$\forall j \in \mathcal{C}, d \in \mathcal{D}: c^{min}_{j,d} \text{ defined} \tag{26}$$
$$\sum_{\substack{e\in\mathcal{E}^{all},\\ s\in\sigma_j:\\ \xi_e\in\Xi_j}} \alpha_{s,\sigma_j}\cdot x_{e,d,s} + f_{j,d} \leq c^{max}_{j,d},$$
$$\forall j \in \mathcal{C}, d \in \mathcal{D}: c^{max}_{j,d} \text{ defined} \tag{27}$$

$$\sum_{\substack{e\in\mathcal{E}^{all},\\ s\in\mathcal{S}_{e,d_1}\cap\sigma_1}} \tilde{x}_{e,d_1,s} = \sum_{\substack{e\in\mathcal{E}^{all},\\ s\in\mathcal{S}_{e,d_2}\cap\sigma_2}} \tilde{x}_{e,d_2,s},$$
$$\forall d_1, d_2 \in \mathcal{D}^{all}, \sigma_1, \sigma_2 \in \mathcal{Z}:$$
$$L^{ward,both}_{d_1,d_2,\sigma_1,\sigma_2} = 1 \tag{28}$$
$$\sum_{\substack{e\in\mathcal{E}^{all},\\ s\in\mathcal{S}_{e,d_1}\cap\sigma_1}} \tilde{x}_{e,d_1,s} \leq \sum_{\substack{e\in\mathcal{E}^{all},\\ s\in\mathcal{S}_{e,d_2}\cap\sigma_2}} \tilde{x}_{e,d_2,s},$$
$$\forall d_1, d_2 \in \mathcal{D}^{all}, \sigma_1, \sigma_2 \in \mathcal{Z}:$$
$$L^{ward,one}_{d_1,d_2,\sigma_1,\sigma_2} = 1 \tag{29}$$

4.2.7 Balancing the work

To balance the workload on different shifts, Equations (30)-(31) bind variables for highest and lowest excess staffing. The difference is then penalized in the objective function.

$$\sum_{\substack{e\in\mathcal{E}^{all},\\ s\in\sigma_j}} \alpha_{s,\sigma_j}\cdot x_{e,d,s} + f_{j,d} - c^{+}_{j} \leq c^{min}_{j,d},$$
$$\forall j \in \mathcal{C}, d \in \mathcal{D}: \Xi_j = \bigcup_{e\in\mathcal{E}^{all}} \xi_e$$
$$c^{min}_{j,d} \text{ defined, } c^{max}_{j,d} \text{ not defined} \tag{30}$$
$$\sum_{\substack{e\in\mathcal{E}^{all},\\ s\in\sigma_j}} \alpha_{s,\sigma_j}\cdot x_{e,d,s} + f_{j,d} - c^{-}_{j} \geq c^{min}_{j,d},$$
$$\forall j \in \mathcal{C}, d \in \mathcal{D}: \Xi_j = \bigcup_{e\in\mathcal{E}^{all}} \xi_e$$
$$c^{min}_{j,d} \text{ defined, } c^{max}_{j,d} \text{ not defined} \tag{31}$$

To spread the assignments of unpopular shifts evenly, Equations (32)-(33) link the variables for the assignments to those shifts, relative to the maximum assignments allowed for the given employee. The objective function then penalizes the difference between the highest and lowest percentage of such assignments.

$$\sum_{\substack{d\in\mathcal{D},\\ s\in\sigma}} \alpha_{s,\sigma}\cdot x_{e,d,s} - \tilde{M}^{total,ub}_{e,\sigma}\cdot\zeta^{max}_{\sigma} \geq 0,$$
$$\forall e \in \mathcal{E}, \sigma \in \mathcal{Z}^{spread} \tag{32}$$
$$\sum_{\substack{d\in\mathcal{D},\\ s\in\sigma}} \alpha_{s,\sigma}\cdot x_{e,d,s} - \tilde{M}^{total,ub}_{e,\sigma}\cdot\zeta^{min}_{\sigma} \leq 0,$$
$$\forall e \in \mathcal{E}, \sigma \in \mathcal{Z}^{spread} \tag{33}$$

4.2.8 Chaperoning

We let $s_c \in \mathcal{S}$ denote the chaperoning shift. Equation (34) ensures that we only schedule the chaperoning shift simultaneously to a chaperone and one trainee. We note that Equation (16) in Section 4.2.3 ensures that we schedule the right number of chaperoning shifts during the rostering horizon.

$$\sum_{e \in \mathcal{E}_c^{train}} x_{e,d,s_c} = x_{c,d,s_c}, \quad \forall c \in \mathcal{E}^{chap}, d \in \mathcal{D} \tag{34}$$

Equations (35)-(36) evaluate how much two employees are working together on a given day. To promote trainees working with their chaperone or other senior nurses, we include a reward in the objective function.

$$g_{e_1,e_2,d,\sigma} \leq \sum_{s \in \sigma} \alpha_{s,\sigma} \cdot x_{e_1,d,s}, \quad \forall (e_1,e_2) \in \mathcal{E}^G, d \in \mathcal{D}, \sigma \in \mathcal{Z}^{together} \tag{35}$$

$$g_{e_1,e_2,d,\sigma} \leq \sum_{s \in \sigma} \alpha_{s,\sigma} \cdot x_{e_2,d,s}, \quad \forall (e_1,e_2) \in \mathcal{E}^G, d \in \mathcal{D}, \sigma \in \mathcal{Z}^{together} \tag{36}$$

4.2.9 New constraints

Equations (37)-(38) count the number of different shift sets assigned to an employee on a set of consecutive days. If this number exceeds a given value, the objective function penalizes the difference.

$$\tilde{x}_{e,d',s} \leq \kappa_{e,d,\sigma}, \quad \forall e \in \mathcal{E}, d \in \mathcal{D}^{all}, d' \in \{d, \ldots, d + D^{diff} - 1\}, \sigma \in \mathcal{Z}^{diff}, s \in \sigma : d + D^{diff} - 1 \in \mathcal{D} \tag{37}$$

$$\sum_{\sigma \in \mathcal{Z}^{diff}} \kappa_{e,d,\sigma} - n^{diff} \leq \kappa_{e,d}^{more} \quad \forall e \in \mathcal{E}, d \in \mathcal{D}^{all} : d + D^{diff} - 1 \in \mathcal{D} \tag{38}$$

Equations (39)-(41) count the number of work sequences on consecutive days that include assignments to a given shift set (i.e., night shifts). The objective function penalizes when this number exceeds a given value.

$$x_{e,d_2,s} + \theta_{e,d_1}^{start} - \sum_{\substack{d \in \mathcal{D}: \\ d_1 \leq d < d_2}} \theta_{e,d}^{end} - 1 \leq \phi_{e,d_1,\sigma}^{start}, \quad \forall e \in \mathcal{E}, d_1 \in \mathcal{D}^{all}, d_2 \in \mathcal{D}^{all}, \sigma \in \mathcal{Z}_e^{seq}, s \in \mathcal{S}_{e,d_2} \cap \sigma : d_1 \leq d_2 \tag{39}$$

$$x_{e,d_1,s} + \theta_{e,d_2}^{end} - \sum_{\substack{d \in \mathcal{D}: \\ d_1 < d \leq d_2}} \theta_{e,d}^{start} - 1 \leq \phi_{e,d_2,\sigma}^{end}, \quad \forall e \in \mathcal{E}, d_1 \in \mathcal{D}^{all}, d_2 \in \mathcal{D}^{all}, \sigma \in \mathcal{Z}_e^{seq}, s \in \mathcal{S}_{e,d_1} \cap \sigma : d_1 \leq d_2 \tag{40}$$

$$\phi_{e,d,\sigma}^{end} + \sum_{\substack{d' \in \mathcal{D}^{all}: \\ d < d' \leq d + D_{e,\sigma}^{seq}}} \phi_{e,d',\sigma}^{start} - \chi_{e,\sigma} \leq \phi_{e,d,\sigma}^{count}, \quad \forall e \in \mathcal{E}, d \in \mathcal{D}^{all}, \sigma \in \mathcal{Z}_e^{seq} : d + D_{e,\sigma}^{seq} \in \mathcal{D} \tag{41}$$

Equations (39) consider all days from the start and to the end of a sequence, including both days, and check whether we assign $e \in \mathcal{E}$ to shift set $\sigma \in \mathcal{Z}_e^{seq}$ on any of those days. If so, then we force $\phi_{e,d,\sigma}^{start} \in [0,1]$ to be one for the start day $d \in \mathcal{D}^{all}$ of the sequence. Equations (40) work similarly for the end of a sequence. Finally, Equations (41) count the number of sequences including shift set $\sigma \in \mathcal{Z}_e^{seq}$ that we have assigned to employee $e \in \mathcal{E}$ on $D_{e,\sigma}^{seq}$ consecutive days, and forces the penalty variable $\phi_{e,d,\sigma}^{count} \geq 0$ to be positive if we exceed the preferred number.

Equations (42)-(44) measure the number of days off assigned before and after assigning work during a given period. The objective function rewards assigning the surrounding days off for given periods.

$$B_{e,q} + q^b \cdot (\tilde{x}_{e,d_1,s_1} + x_{e,d_2,s_2}) \leq 2q^b + \psi_{d_1,d_2,s_1,s_2},$$
$$\forall e \in \mathcal{E}, q \in \mathcal{Q}_e, d_1 \in \left[q^s - q^b, q^s - 1\right],$$
$$s_1 \in \mathcal{S}_{e,d_1} \cap \mathcal{S}^{work}, d_2 \in \mathcal{D},$$
$$s_2 \in \mathcal{S}_{e,d_2} \cap \mathcal{S}^{work} : d_1 \leq d_2,$$
$$(s_1, d_1) \notin \mathcal{O}_{e,q}, (s_2, d_2) \in \mathcal{O}_{e,q},$$
$$0 \leq \psi_{d_1,d_2,s_1,s_2} < q^b \tag{42}$$

$$A_{e,q} + q^a \cdot (\tilde{x}_{e,d_1,s_1} + x_{e,d_2,s_2}) \leq 2q^a + \psi_{d_1,d_2,s_1,s_2},$$
$$\forall e \in \mathcal{E}, q \in \mathcal{Q}_e, d_2 \in [q^e + 1, q^e + q^a],$$
$$s_1 \in \mathcal{S}_{e,d_1} \cap \mathcal{S}^{work}, d_1 \in \mathcal{D}^{all},$$
$$s_2 \in \mathcal{S}_{e,d_2} \cap \mathcal{S}^{work} : d_1 \leq d_2,$$
$$(s_1, d_1) \in \mathcal{O}_{e,q}, (s_2, d_2) \notin \mathcal{O}_{e,q},$$
$$0 \leq \psi_{d_1,d_2,s_1,s_2} < q^a \tag{43}$$

$$(q^b + q^a) \cdot \sum_{\substack{d \in \mathcal{D}^{all}, \\ s \in \mathcal{S}_{e,d} \cap \mathcal{S}^{work}: \\ (s,d) \in \mathcal{O}_{e,q}}} \tilde{x}_{e,d,s} \geq B_{e,q} + A_{e,q},$$
$$\forall e \in \mathcal{E}, q \in \mathcal{Q}_e \tag{44}$$

Equations (42) ensure that if we assign employee $e \in \mathcal{E}$ to shift $s_2 \in \mathcal{S}^{work}$ during the period $q \in Q_e$ and also to shift $s_1 \in \mathcal{S}^{work}$ less than q^b days before the period, then the variable $B_{e,q}$ is bounded from above by the number of full days off between the two assignments. Similarly, Equations (43) ensure that if we assign employee $e \in \mathcal{E}$ to shift $s_1 \in \mathcal{S}^{work}$ during the period $q \in Q_e$ and also to shift $s_2 \in \mathcal{S}^{work}$ less than q^a days after the period, then the variable $A_{e,q}$ is bounded from above by the number of full days off between the two assignments. Finally, Equations (44) ensure that the variables $B_{e,q}$ and $A_{e,q}$ can only be positive if we assign a work shift to employee $e \in \mathcal{E}$ during period $q \in \mathcal{Q}_e$.

4.3 Objective function

The objective function is a linear combination of penalties and rewards related to the violation or satisfaction of soft constraints. Table 11 presents the notation for objective weights.

We let Ω_i for $i \in \{1, \ldots, 9\}$ represent different terms in the objective function. The first objective term Ω_1 relates to the satisfaction and violation of preferences.

$$\Omega_1 = \sum_{(e,d,s) \in \mathcal{A}} \omega^{assign}_{e,d,s} \cdot x_{e,d,s} \tag{45}$$

The second objective term Ω_2 penalizes when exceeding the preferred maximum number of consecutive days without a protected day off.

$$\Omega_2 = \sum_{\substack{e \in \mathcal{E}, \\ d \in \mathcal{D}^{all}}} \omega^{pf} \cdot v_{e,d} \tag{46}$$

The third objective term Ω_3 penalizes for deviating from targets for the assigned hours, both for the rostering horizon as a whole and for specific weeks.

$$\Omega_3 = \sum_{e \in \mathcal{E}} \left(\omega^{+} \cdot t^{+}_e + \omega^{-} \cdot t^{-}_e + \omega^{++} \cdot t^{++}_e + \omega^{--} \cdot t^{--}_e\right) + \sum_{\substack{e \in \mathcal{E}, \\ w \in \mathcal{W}}} \omega^{week} \cdot \left(\lambda^{+}_{e,w} + \lambda^{-}_{e,w}\right) \tag{47}$$

The fourth objective term Ω_4 penalizes when a series of consecutive work assignments is either shorter than the preferred minimum or longer than the preferred maximum.

$$\Omega_4 = \sum_{\substack{e \in \mathcal{E}, \\ d \in \mathcal{D}}} \omega^{seqlen} \cdot \delta_{e,d} + \sum_{\substack{e \in \mathcal{E}, \\ d \in \mathcal{D}^{all}, \\ \sigma \in \mathcal{Z}}} \omega^{maxrow}_e \cdot \mu_{e,d,\sigma} \tag{48}$$

The fifth objective term Ω_5 penalizes when assigning unwanted series of assignments.

$$\Omega_5 = \sum_{\substack{e \in \mathcal{E}, \\ d \in \mathcal{D}^{all}, \\ y \in \mathcal{Y}}} \omega^{pat}_y \cdot \pi_{e,d,y} \tag{49}$$

The sixth objective term Ω_6 relates to the coverage constraints. It penalizes when not reaching the preferred coverage along with penalizing the use of float nurses.

$$\Omega_6 = \sum_{\substack{j \in \mathcal{C}, \\ d \in \mathcal{D}}} \omega^{nonmax}_j \cdot \left(c^{max}_{j,d} - \sum_{\substack{e \in \mathcal{E}^{all}, \\ s \in S_{e,d} \cap \sigma_j}} \alpha_{s,\sigma_j} \cdot x_{e,d,s} \right) + \sum_{\substack{j \in \mathcal{C}, \\ d \in \mathcal{D}}} \omega^{float}_j \cdot f_{j,d} \tag{50}$$

The seventh objective term Ω_7 contributes towards balancing the work. It penalizes for an uneven distribution of excess staffing along with an uneven distribution of unpopular shifts.

$$\Omega_7 = \sum_{j \in \mathcal{C}} \omega^{bal} \cdot \left(c^{+}_j - c^{-}_j\right) + \sum_{\sigma \in \mathcal{Z}^{spread}} \omega^{spread}_\sigma \cdot \left(\zeta^{max}_\sigma - \zeta^{min}_\sigma\right) \tag{51}$$

Table 11 Weights used in the objective function

Weight	Description
$\omega^{assign}_{e,d,s}$	Penalty or reward for assigning employee $e \in \mathcal{E}$ to shift $s \in \mathcal{S}$ on day $d \in \mathcal{D}$.
ω^{pf}	Penalty for exceeding the maximum distance between two contagious protected days off.
ω^{+}	Penalty for a positive deviation from the target hours, within the bound N^{+}.
ω^{++}	Penalty for a positive deviation from the target hours, exceeding N^{+} .
ω^{-}	Penalty for a negative deviation from the target hours, within the bound N^{-}.
ω^{--}	Penalty for a negative deviation from the target hours, exceeding N^{-}.
ω^{week}	Penalty for deviation from the weekly targets.
ω^{seqlen}	Penalty for violating the minimum sequence.
ω^{maxrow}_{e}	Penalty for exceeding the number of maximum assignments in a row for employee $e \in \mathcal{E}$.
ω^{pat}_{y}	Penalty for assigning non-preferred pattern $y \in \mathcal{Y}$.
ω^{float}_{j}	Penalty for assigning a float nurse to coverage $j \in \mathcal{C}$.
ω^{nonmax}_{j}	Penalty for not hitting the maximum target for coverage $j \in \mathcal{C}$.
ω^{bal}	Penalty for not balancing the excess personnel.
ω^{spread}_{σ}	Penalty for not spreading assignments to shift set $\sigma \in \mathcal{Z}^{spread}$ relatively equally between the employees.
$\omega^{together}_{e_1,e_2}$	Reward for assigning employees $(e_1, e_2) \in \mathcal{E}^{G}$ work together.
ω^{diff}	Penalty for the number of shift sets exceeding the preferred we assign on D^{diff} consecutive days.
$\omega^{seqconsec}$	Penalty for exceeding the maximum number of sequences including a certain shifts on consecutive days.
ω^{b}	Reward for scheduling surrounding days off before a period where it is preferred.
ω^{a}	Reward for scheduling surrounding days off after a period where it is preferred.

The eight objective term Ω_8 provides a reward when trainees work together with their chaperones or other senior nurses.

$$\Omega_8 = \sum_{\substack{(e_1,e_2)\in\mathcal{E}^G,\\ d\in\mathcal{D},\\ \sigma\in\mathcal{Z}^{together}}} \omega^{together}_{e_1,e_2} \cdot g_{e_1,e_2,d,\sigma} \tag{52}$$

The last term of the objective function Ω_9 relates to the new constraints. First, it penalizes for assigning too many different shifts on a series of consecutive days. Second, it penalizes for too many work sequences that include a specific shift on consecutive days. Third, it rewards for assigning surrounding days off for some work periods.

$$\begin{aligned}\Omega_9 = &\sum_{\substack{e\in\mathcal{E},\\ d\in\mathcal{D}^{all}}} \omega^{diff} \cdot \kappa^{more}_{e,d} \\ &+ \sum_{\substack{e\in\mathcal{E},\\ d\in\mathcal{D}^{all},\\ \sigma\in\mathcal{Z}^{seq}_{e}}} \omega^{seqconsec} \cdot \phi^{count}_{e,d,\sigma} \\ &+ \sum_{\substack{e\in\mathcal{E},\\ q\in\mathcal{Q}}} \left(\omega^{b} \cdot B_{e,q} + \omega^{a} \cdot A_{e,q}\right)\end{aligned} \tag{53}$$

The overall objective minimizes a linear combination of these nine terms, as given with Equation (54).

$$\min \sum_{i\in\{1,\ldots,9\}} \Omega_i \tag{54}$$

4.4 Comparison to other formulations

This section has introduced a flexible MIP formulation of the problem described in Section 3. We will now discuss specific aspects of the formulation that differ from other formulations in the literature.

Table 12 compares this formulation to four other nurse rostering formulations that have employed integer programming approaches. We compare these formulations across a wide range of criteria, most relating to either the framework around the model or the constraints included. In general, the framework we present is more flexible than those used in previous formulations. Furthermore, we include most of the constraints present in previous formulations.

Many MIP formulations are assignment-based, i.e., the main decision variables are binary corresponding to whether a nurse is assigned to a specific day and shift. Valouxis et al. (2012) break the problem down into smaller sub-problems that are assignment-based and solve them sequentially. This approach is only applicable for formulations with a few hard constraints of a specific structure, where combining feasible solutions to all sub-problems directly results in an overall feasible solution. Due to the number and structure of hard constraints in the formulation we present, this type of decomposition would likely not be successful.

Burke and Curtois (2014) decompose the problem and formulate the pricing problems as network flow problems, more specifically resource constrained shortest paths. They generate the paths (i.e., pricing problem solutions) dynamically and prune dominated paths

Table 12 A comparison to other formulations that have used MIP-based approaches to formulate the NRP.

	This formulation	Burke and Curtois (2014)	Mischek and Musliu (2017)	Rönnberg and Larsson (2010)	Valouxis et al. (2012)
Formulation origin	Real case from Denmark	A collection of benchmarks and INRC-I	Multi-stage formulation (INRC-II with extensions)	Real case from Sweden	INRC-I
MIP approach	Assignment based	Branch and price	Assignment based	Assignment based	Two-phase algorithm
Definition of shifts					
• corresponding to coverage	✓	✓	✓	✓	✓
• other work shifts	✓			✓(fixed prior)	
• days off	✓				
• grouping	✓				
Skills	✓	✓	✓	✓	✓
Requests	✓	✓	✓	✓	✓
Distinction between types of days off	Protected days off, parental leave, annual leave, public holidays and other days off	None	None	Holidays (fixed prior) and other days off	None
Structure of coverage constraints	Shift set	Shift type	Shift type	Shift type	Shift type
Daily restrictions	Using conflict cliques	Max one shift a day	Max one shift a day	Max one shift a day	Max one shift a day
Restricting shift successions	✓(using conflict cliques)	✓	✓	✓	✓
Restrictions on total assignments					
• min/max shifts		✓	✓		✓
• min/max hours	✓			✓	
• min/max shift type	✓	✓			
Restrictions on assignments within a specific period (e.g., weeks)					
• min/max shifts		✓	✓		
• min/max hours	✓			✓	
• min/max shift type	✓				
Consecutive restrictions					
• min work days	✓	✓	✓		✓
• max work days	✓	✓	✓	✓	✓
• min shift type		✓	✓		
• max shift type	✓	✓	✓		
• min days off		✓	✓		✓
• max days off	✓	✓	✓		✓
• max work hours on a set of days	✓				
• max different shift types on a set of days	✓				
• max work sequences with a shift type on a set of days	✓				
• unwanted patterns	✓	✓			✓
Weekend restrictions					
• complete weekends / weekend patterns	✓	✓	✓	✓	✓
• min/max total weekends worked		✓	✓		✓
• min/max consecutive weekends worked		✓			✓
• even distribution				✓	
• surrounding days off	✓				
Balancing					
• excess coverage	✓				
• some shift types	✓			✓	
Chaperoning	✓				

as they go through the graph. They state that dominance checking is the most time-consuming part of this approach.

We believe that a network flow based approach would not be successful in addressing the formulation we present, for example, as the flexibility of the protected days off makes it difficult to define clear domination criteria. Furthermore, allowing more than one shift a day requires additional arcs, changing the structure of the graph. Finally, we include constraints for chaperoning that require two nurses assigned to the same shift simultaneously and an assignment based formulation provides a better overview when assigning these shifts.

Regarding shifts, the standard in previous formulations is a one-to-one mapping between shifts and coverage requirements. This is evident by both nurse rostering competitions (Haspeslagh et al., 2014; Ceschia et al., 2019) and numerous benchmarks (Burke and Curtois, 2014; Curtois and Qu, 2014). While Rönnberg and Larsson (2010) also include other work shifts, their timing is fixed and the roster is generated around them. The formulation we have presented is capable of scheduling these shifts (e.g., administrative work) when it fits best with respect to the roster as a whole. If needed, these shifts may be fixed with the request environment.

We also model days off as shift types and distinguish between different types of days off, which is uncommon in the literature. While Rönnberg and Larsson (2010) differentiate between holidays and other days off, they assume that the holidays are fixed and pre-approved by the head nurse.

Due to the extended definition of shifts, we do not explicitly model many of the constraints regarding administrative work (e.g., meetings or courses) or days off. Instead, we set parameters corresponding to those shifts and express the constraints in a general manner. As an example, Equation (16) ensures that we assign the correct number of protected days off during the rostering horizon. Similarly, Equation (17) ensures that some employees get administrative shifts each week.

The legislation regarding protected days off in Denmark is quite complex. The literature includes some simple constraints regarding days off (either explicitly or implicitly). Those constraints consider all days off to be the same, independent of the previous work shift. Thus, any distinction between days off following day shifts, evening shifts, or night shifts is presented using working patterns that require binary variables (similar to Equation (23)).

Equations (6)-(10) present flexible constraints to control the requirements for protected days off by tracking adjacent work shifts. These constraints can be adjusted to fit different requirements by simply modifying the parameter ρ_i^{pf}. Furthermore, they provide the freedom to decide how many protected days off should be scheduled in a row for each occurrence. These constraints, together with Equations (11) and (16) capture the complex Danish legislation.

While most formulations define coverage constraints based on single shift types, we define them based on shift sets. These sets allow for partial coverage and different combinations of shift types when composing the coverage.

Due to the extended definition of shifts, we replace the popular constraints for a maximum of one shift a day with the opposite, namely a minimum of one shift a day. To prevent physically impossible or illegal combinations of shifts we automatically generate conflict cliques, where each clique represents a set of assignments where at most one may be scheduled (as given with Equation (2)).

In this formulation, we do not explicitly indicate which shifts are conflicting but instead introduce parameters (ρ^{full} and ρ^{red}) describing the minimum time off between consecutive work shifts. We use these parameters to build the conflict graph and generate the conflict cliques. Thus, we do not generate redundant constraints that will likely not be binding. By automatically generating conflict cliques, we obtain increased flexibility and are able to capture different restrictions, e.g., allowing multiple shifts a day. At the same time, we do not weaken the formulation as we would by pairwise modeling of all conflicting shifts on a given day.

The conflict cliques also capture forbidden shift successions, producing the tightest possible constraints. Previous formulations have often formulated forbidden shift successions based on forbidding certain pairs (Curtois and Qu, 2014; Mischek and Musliu, 2017; Rönnberg and Larsson, 2010).

Santos et al. (2016) noted the problem structure, namely that it included multiple constraints with an upper bound of one. To strengthen the IP formulation, they employed clique cuts similar to the ones we generate by drawing upon parts of the constraint matrix.

Many formulations define most or all constraints based on a number of shifts. While we define several constraints in that manner, we also define many constraints based on hours. Generally, contracts are based on hours, and when the shifts differ in length, using hours in the formulation results in more accurate constraints.

Equations (24)-(25) for assignments scheduled in a combination with other assignments provide a flexible alternative to model some working patterns. By modifying parameters ($L^{emp,both}_{e,d_1,d_2,\sigma_1,\sigma_2}$ or $L^{emp,one}_{e,d_1,d_2,\sigma_1,\sigma_2}$) we can easily tailor these constraints to the need of each em-

ployee. For example, we can relax the requirement for scheduling complete weekends for a specific employee and a specific weekend. Moreover, we can define different weekend patterns for different employees. In the wards we collaborated with, weekends off are scheduled long in advance. Therefore, we do not include constraints regarding the allocation of weekend work (e.g., a maximum number of weekends or an even distribution of weekends).

Section 4.2.9 introduced three new constraints regarding assignments to individual employees on consecutive days. In theory, we could formulate these constraints using Equation (23) for unwanted series. Nevertheless, such a formulation would have two severe disadvantages.

First, to use Equation (23) we would need to enumerate all combinations of patterns. This enumeration would not be trivial for any of the new constraints and defining them using patterns would reduce the transparency when communicating the model and the results to practitioners and employees. Furthermore, such enumeration is not flexible if the ward would make any changes, e.g., adding a new shift type.

Second, all of these new constraints are soft constraints and thus require variables to be penalized in the objective function. If we were to formulate these constraints as patterns, we would introduce an abundance of binary variables corresponding to each specific pattern. However, by formulating them explicitly with Equations (37)-(44) we suffice with using continuous variables. Therefore, using working patterns would not only increase the complexity of the input but also the model.

We note that formulating constraints using regular expressions as Burke and Curtois (2014) would require a binary variable for each and every constraint (i.e., either we have a match for the expression or not). While that is not an issue in their approach, transferring the formulation to an assignment based MIP model would likely be unsuccessful.

5 Results

This section presents the results of implementing this rostering system in practice in two wards. Section 5.1 starts by describing the wards and introducing the instances. Then, Section 5.2 presents the computational results from utilizing a commercial solver to solve the MIP model (see electronic appendix). Finally, Section 5.3 discusses the impact of the rostering system in practice, including feedback from the practitioners.

5.1 Instances

During this research, we collaborated closely with two wards in Region Zealand. The wards have different specialties and are located in different hospitals. The main similarity between the two wards is that they both use a rostering horizon of four weeks. We refer to these wards as *Ward A* and *Ward B*.

The specialty in Ward A requires a long education, part of which is performed within the ward. Furthermore, nurses with that specialty are a very scarce resource and the manager strives to make rosters to their liking. Therefore, we impose no limitations on how many requests each employee can have, resulting in an average of 10 requests per employee for a 4-week rostering horizon. Furthermore, the employees' attitude towards the request environment differs, where some employees consistently input multiple requests while others are more flexible, perhaps putting forward a single request.

The request culture in Ward B is significantly different, where the average number of requests per employee range from 0.3 to 2.2 for the entire rostering horizon, with the only exception being around Christmas. This difference is not because of any limitations set by the system, but rather due to the current practices in the wards. In Ward A, the employees are used to having a large influence on their rosters, whereas those in Ward B are used to the manager creating the schedules.

In this paper, we report the results for 12 different instances available from Böðvarsdóttir et al. (2019), where instances A01-A07 come from Ward A and instances B01-B05 from Ward B. Table 13 presents some summary statistics for the different instances. The first two columns show the number of employees, both those that we should generate a full roster for and all employees in the ward (including those that have fixed rosters or only partial rosters). The third column shows the number of feasible assignments for the instance, referring to all feasible combinations of employee, day and shift type. The fourth and fifth columns compare the number of conflicting pairs of assignments with the number of conflict cliques.

From the table, we can see that the instances for Ward A are larger than for Ward B. Furthermore, using conflict cliques to prohibit physically infeasible or illegal combinations of assignments significantly reduces the number of constraints needed to prevent conflicting assignments (Equation (2) in Section 4.2.1). Comparing the last two columns shows that the average reduction is above a factor of 50.

In general, the clique constraints presented with Equation 2 make up less than 5% of the total constraints of

Table 13 Summary statistics for the twelve instances.

Instance	Employees $\mathcal{E}$	Employees $\mathcal{E}^{all}$	Feasible assignments	Conflicts pairs	Conflicts cliques
A01	45	48	4,809	141,666	2,285
A02	45	47	4,764	120,846	2,322
A03	47	50	4,516	122,230	2,233
A04	47	49	4,976	149,726	1,970
A05	46	48	5,158	162,106	2,123
A06	46	49	4,638	136,111	2,401
A07	46	48	4,504	135,675	2,296
B01	33	46	3,846	65,392	1,841
B02	31	44	3,565	79,297	1,314
B03	30	44	3,607	63,000	1,652
B04	38	43	3,930	67,399	1,816
B05	40	42	4,077	69,946	1,790

the model. If we were to replace these constraints with the forbidding all pairs of conflicting assignments, the corresponding constraints would make up around 70% of the alternative model.

5.2 Computational results

We solved the MIP model with Gurobi 8.0 using the default parameter settings and a time limit of 1 hour. We conducted all experiments on a 64-bit Windows 7 with 12GB RAM and an Intel Core i5-4570 CPU @3.20GHz. Table 14 presents the computational results.

Table 14 Computational results.

Instance	Objective value	MIP Gap (%)	Time (s)
A01	-3,358,119.41	0.00	855.46
A02	-4,262,959.81	0.00	1,026.45
A03	-4,230,378.72	0.00	374.00
A04	-2,775,710.80	0.00	362.13
A05	-3,718,385.02	0.13	3,600.00
A06	-2,542,988.57	0.00	186.61
A07	-4,241,033.51	0.09	3,600.00
B01	22,879,748.33	0.00	274.73
B02	34,275,500.56	0.00	42.85
B03	21,992,210.67	0.00	151.39
B04	16,982,883.97	0.00	85.04
B05	60,713,240.41	0.01	3,600.00

We solve most instances to optimality, i.e., within Gurobi's default MIP gap of 10^{-4}. For three instances, we terminate due to time, but with a very low gap. For more than half of the instances, we can prove optimality in around 5 minutes. For the remaining instances, we reach a small gap (i.e., below 1%) rather early and then most of the effort is spent on improving the bound.

To further draw out the strength of this formulation, we compare the solutions found to the objective values of the LP relaxations. This comparison shows an LP-IP gap around 2-5% for most instances. Only two instances have a gap above 6%, i.e., A06 with a 21% gap and B04 with an 11% gap. Still, we note that both instances were solved to optimality within less than 200 seconds.

These results show that the formulation of sensible MIP models, with a limited number of integer variables and a tight LP-relaxation, can lead to good results for practical problems.

5.3 Practical impact

Practitioners are neither concerned with the objective value of a solution nor the MIP gap, they are only interested in the roster itself. Therefore, we generally run the optimization for a short time (e.g., 5 minutes) to generate good rosters, as opposed to spending time on proving optimality.

As discussed in Section 3.3, we tune the objective weights for each instance to obtain a high-quality roster. When tuning the weights we evaluate multiple components including specific requests, deviation from contractual hours, the need for float nurses, the number of days between protected days off, etc.

For increased transparency, we do not only report the shift assignments but also include various performance indicators. These indicators provide a better overview and give the managers a better chance of fairly assessing the quality of the roster. Table 15 presents an example of performance indicators for the 12 instances, namely the number of float nurses needed along with the percentage of fulfilled requests. Although multiple factors come into play when assessing the quality of the roster, these two are the most important to the practitioners.

Table 15 Number of float nurses required along with the fulfillment of requests.

Instance	**Float nurses**	**Fulfilled requests (%)** High priority	All requests
A01	0	98.61	89.29
A02	0	98.81	90.40
A03	0	97.96	89.92
A04	0	93.68	82.33
A05	2	97.31	89.33
A06	0	97.70	90.50
A07	0	98.27	85.83
B01	8	62.50	75.00
B02	6	66.67	74.26
B03	12	100.00	100.00
B04	12	100.00	93.53
B05	34	100.00	97.38

We can see that the use of float nurses differs significantly between the two wards. Ward A rarely requires float nurses to satisfy their staffing requirements, with the only exceptions being when many of the permanent staff are on holiday, e.g., New Year's Day. However, all instances for Ward B require float nurses to be able to meet the staffing requirements in the weekends. Furthermore, instance B05 requires additional float nurses to meet the staffing requirements for nights on some weekdays.

The fulfillment of requests also differs between the wards, mostly due to the different number of requests. Overall, we manage to satisfy a majority of the requests for all instances. We acknowledge that instances B01 and B02 stand out, with rather low percentages. Nonetheless, these percentages are close to the best possible for both instances, as requests conflicting with hard constraints make it impossible to fulfill more.

Due to lack of data, we are unable to directly compare these results to manually generated rosters. Before implementing the rostering system, only the shifts actually worked were stored in a salary system. Disruptions (e.g., sick leave) may cause changes in the original roster and those changes were not documented. Furthermore, the nurses' requests were gathered on a physical sheet (or sometimes only verbal communication) and were not specifically stored after the generation of a roster. Therefore, qualitative feedback from the users is the only way to evaluate the quality of the rosters generated and the practical impact of this system.

The practitioners are very pleased with the rostering system and declare that it is beneficial, both for the employees and the patients. In general, the managers find the process of generating rosters much easier, while the results are of higher quality. As an example, the MIP model produces rosters that better utilize the personnel's contractual hours while simultaneously meeting their individual needs.

Overall, the employees are happy with the rostering system, and managers have applauded the increased fairness that it has brought. Furthermore, the employees have an overview of the requests and fixed assignments for the entire ward, which has resulted in increased transparency and improved morale. At last, the managers can trust that all rosters produced with the model satisfy the complex legislative rules, significantly reducing the time they need to spend on validation.

6 Conclusion

In this paper, we have presented a flexible nurse rostering system and a comprehensive MIP formulation that we have implemented in two wards in Danish hospitals. In addition to introducing new constraints, we also provide flexible alternatives for common constraints from the literature. The system has been developed in collaboration with experienced managers to match the complex needs of real hospitals. The system's adaptability is a major advantage, as it can be special-tailored to fit the needs of a specific ward, along with the needs of individual employees. This research has already brought significant improvement to the scheduling procedures in Danish hospitals.

Due to the close collaboration with practitioners, we have obtained some insights that we believe will be beneficial for other researchers. Although most research on nurse rostering focuses on mathematical aspects, e.g., solution algorithms, the greatest challenges we faced when implementing this system were not mathematical. When entering a ward that uses manual rostering, a large part of the process is unveiling different elements that the managers take into account when creating rosters. Obviously, the overall legislation along with individual contracts set clear guidelines. Nonetheless, the managers consider multiple softer aspects, such as preferences for shifts or unwanted patterns of assignments, and bringing all these aspects to the surface is an iterative process.

We can attest that an open dialogue with practitioners throughout the development is crucial to capture their needs. In our experience, practitioners may leave out some of the critical aspects early on, for example, aspects that they intuitively but unknowingly consider when manually generating rosters. On the contrary, some of the constraints that they set early on turned out to be of a low priority when assessing the quality of the roster, as is reflected in the objective weights. Thus, ensuring that we address the right problem for implementation may be an underrated challenge.

Solving optimization problems involving human resources is an intricate task and has to be treated with respect. To develop solutions that can be adopted in practice, we need to fully grasp the needs of practitioners. As an example, we would not have recognized the immense flexibility required without working closely with practitioners and addressing the exact problem they face. Hospital rostering is conducted in a dynamic environment, and we need to build solutions that can assist managers in creating good rosters across a wide range of different scenarios.

During this research, we have identified several opportunities that may further contribute to getting nurse rostering research implemented in practice. A recurring challenge throughout this project has been the sensitiv-

ity of the objective function with respect to the input data and the objective weights. As the data differs significantly between rostering horizons, we need to tune the weights for each instance to arrive at good rosters. This process is currently manual, and even though the overall resources needed to create each roster has been significantly reduced, the need for this type of manual interventions reduces the advantages of using automated methods.

Furthermore, objective weights are an abstract concept where the impact of a particular weight (or combination of weights) may not be straightforward. As practitioners often have a limited mathematical background, complex models and solution methods are often inaccessible to them. Therefore, they may be hesitant to adopt such methods and question the resulting rosters, especially if they are different from the rosters they would have produced manually. In general, practitioners would like to understand the reason for different violations in order to increase their trust in the system. Thus, to bridge the gap between academia and practice, we need to find common ground for communication, independent of mathematical abilities.

Acknowledgements We thank the Department of Data and Development Support at Region Zealand (DU) for partnering with us on this project, along with the Danish Ministry of Health for providing with funds. We especially thank Lena Marie Fredholm Jensen, Allan Bo Hansen and Anne Bernth at DU, along with Joël Raucq at Raucq Consulting for their contribution to the project. Additionally, we thank the wards we have worked with for invaluable insight and feedback along with providing us with data. Finally, we thank Professor David Pisinger along with anonymous reviewers for their feedback that helped us improve this manuscript.

References

Agyei W, Obeng-Denteh W, Andaam EA (2015) Modeling nurse scheduling problem using 0-1 goal programming: A case study of tafo government hospital, kumasi-ghana. International Journal of Scientific & Technology Research 4(3):5–10

Azaiez MN, Al Sharif SS (2005) A 0-1 goal programming model for nurse scheduling. Computers and Operations Research 32(3):491–507, DOI 10.1016/S0305-0548(03)00249-1

Baker KR (1974) Scheduling a full-time workforce to meet cyclic staffing requirements. Manage Sci 20(12):1561–1568

Bard JF, Purnomo HW (2005) Preference scheduling for nurses using column generation. European Journal of Operational Research 164(2):510–534, DOI 10.1016/j.ejor.2003.06.046

Beddoe G, Petrovic S, Li J (2009) A hybrid metaheuristic case-based reasoning system for nurse rostering. Journal of Scheduling 12(2):99–119, DOI 10.1007/s10951-008-0082-8

Bilgin B, De Causmaecker P, Rossie B, Vanden Berghe G (2012) Local search neighbourhoods for dealing with a novel nurse rostering model. Annals of Operations Research 194(1):33–57, DOI 10.1007/s10479-010-0804-0

Burke EK, Curtois T (2014) New approaches to nurse rostering benchmark instances. European Journal of Operational Research 237(1):71–81, DOI 10.1016/j.ejor.2014.01.039, URL http://dx.doi.org/10.1016/j.ejor.2014.01.039

Burke EK, Cowling P, De Causmaecker P, Vanden Berghe G (2001) A memetic approach to the nurse rostering problem. Applied Intelligence 15(3):199–214, DOI 10.1023/A:1011291030731

Burke EK, De Causmaecker P, Vanden Berghe G, Van Landeghem H (2004) The state of the art of nurse rostering. Journal of Scheduling 7(6):441–449, DOI 10.1023/B:JOSH.0000046076.75950.0b

Burke EK, De Causmaecker P, Petrovic S, Vanden Berghe G (2006) Metaheuristics for handling time interval coverage constraints in nurse scheduling. Applied Artificial Intelligence 20(9):743–766, DOI 10.1080/08839510600903841

Burke EK, Li J, Qu R (2010) A hybrid model of integer programming and variable neighbourhood search for highly-constrained nurse rostering problems. European Journal of Operational Research 203(2):484–493, DOI 10.1016/j.ejor.2009.07.036

Burns R, Koop G (1987) A modular approach to optimal multiple-shift manpower scheduling. Operations Research 35(1):100–110

Böðvarsdóttir EB, Bagger NCF, Høffner LE, Stidsen T (2019) Data for research on nurse rostering in Denmark [Data set]. https://doi.org/10.5281/zenodo.3966788

Ceschia S, Dang N, De Causmaecker P, Haspeslagh S, Schaerf A (2019) The second international nurse rostering competition. Annals of Operations Research 274(1-2):171–186

Christiansen ML, Petersen NC, Range TM (2014) Automatiseret vagtplanlægning for sundhedspersonale [Automatic scheduling for healthcare personnel]. In: Fremtidens Hospital, Munksgaard, chap 25, pp 375–387

Curtois T, Qu R (2014) Computational results on new staff scheduling benchmark instances. Technical report

De Causmaecker P, Vanden Berghe G (2011) A categorisation of nurse rostering problems. Journal

of Scheduling 14(1):3–16, DOI 10.1007/s10951-010-0211-z

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

Dohn A, Mason A (2013) Branch-and-price for staff rostering: An efficient implementation using generic programming and nested column generation. European Journal of Operational Research 230(1):157–169, DOI 10.1016/j.ejor.2013.03.018

Gärtner J, Bohle P, Arlinghaus A, Schafhauser W, Krennwallner T, Widl M (2018) Scheduling matters-some potential requirements for future rostering competitions from a practitioner's view. In: PATAT 2018 - Proceedings of the 12th International Conference on the Practice and Theory of Automated Timetabling, pp 33–42

Glass CA, Knight RA (2010) The nurse rostering problem: A critical appraisal of the problem structure. European Journal of Operational Research 202(2):379–389, DOI 10.1016/j.ejor.2009.05.046

Haspeslagh S, De Causmaecker P, Schaerf A, Stølevik M (2014) The first international nurse rostering competition 2010. Annals of Operations Research 218(1):221–236, DOI 10.1007/s10479-012-1062-0

Jensen L, Horsted CP, Lunde A, Hansen MB (2008) Vagtplanlægning i det danske sygehusvæsen [Employee scheduling in the Danish healthcare sector]

Kellogg DL, Walczak S (2007) Nurse scheduling: From academia to implementation or not? Interfaces 37(4):353–369, DOI 10.1287/inte.1060.0247

Lin CC, Kang JR, Liu WY, Deng DJ (2014) Modelling a nurse shift schedule with multiple preference ranks for shifts and days-off. Mathematical Problems in Engineering 2014:10–13, DOI 10.1155/2014/937842

Lin CC, Kang JR, Chiang DJ, Chen CL (2015) Nurse Scheduling with Joint Normalized Shift and Day-Off Preference Satisfaction Using a Genetic Algorithm with Immigrant Scheme. International Journal of Distributed Sensor Networks 2015, DOI 10.1155/2015/595419

Maenhout B, Vanhoucke M (2010) Branching strategies in a branch-and-price approach for a multiple objective nurse scheduling problem. Journal of Scheduling 13(1):77–93, DOI 10.1007/s10951-009-0108-x

Mihaylov M, Smet P, Van Den Noortgate W, Vanden Berghe G (2016) Facilitating the transition from manual to automated nurse rostering. Health Systems 5(2):120–131, DOI 10.1057/hs.2015.12

Mischek F, Musliu N (2017) Integer programming model extensions for a multi-stage nurse rostering problem. Annals of Operations Research pp 1–21, DOI 10.1007/s10479-017-2623-z

Petrovic S (2019) "you have to get wet to learn how to swim" applied to bridging the gap between research into personnel scheduling and its implementation in practice. Annals of Operations Research 275(1):161–179

Rahimian E, Akartunalı K, Levine J (2017) A hybrid integer programming and variable neighbourhood search algorithm to solve nurse rostering problems. European Journal of Operational Research 258(2):411–423, DOI 10.1016/j.ejor.2016.09.030

Rigsrevisionen (2015) Beretning til Statsrevisorerne om hospitalernes brug af personaleresurser [Report to the state auditors on the hospitals use of human resources]

Römer M, Mellouli T (2016) A direct milp approach based on state-expanded network flows and anticipation for multi-stage nurse rostering under uncertainty. In: Proceedings of the 11th international conference on the practice and theory of automated timetabling, pp 549–551

Rönnberg E, Larsson T (2010) Automating the self-scheduling process of nurses in swedish healthcare: A pilot study. Health Care Management Science 13(1):35–53, DOI 10.1007/s10729-009-9107-x

Rönnberg E, Larsson T, Bertilsson A (2013) Automatic scheduling of nurses: What does it take in practice? Springer Optimization and Its Applications 74:151–178, DOI 10.1007/978-1-4614-5094-8_8

Santos HG, Toffolo TA, Gomes RA, Ribas S (2016) Integer programming techniques for the nurse rostering problem. Annals of Operations Research 239(1):225–251, DOI 10.1007/s10479-014-1594-6

Smet P, Brucker P, De Causmaecker P, Vanden Berghe G (2016) Polynomially solvable personnel rostering problems. European Journal of Operational Research 249(1):67–75, DOI 10.1016/j.ejor.2015.08.025

Smet P, Salassa F, Vanden Berghe G (2017) Local and global constraint consistency in personnel rostering. International Transactions in Operational Research 24(5):1099–1117, DOI 10.1111/itor.12357

Stølevik M, Nordlander TE, Riise A, Frøyseth H (2011) A hybrid approach for solving real-world nurse rostering problems. Lecture Notes in Computer Science (including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 6876:85–99, DOI 10.1007/978-3-642-23786-7_9

Turhan AM, Bilgen B (2020) A hybrid fix-and-optimize and simulated annealing approaches for nurse rostering problem. Computers and Industrial Engineering 145:106531, DOI 10.1016/j.cie.2020.106531

Valouxis C, Gogos C, Goulas G, Alefragis P, Housos E (2012) A systematic two phase approach for the nurse rostering problem. European Journal of Operational Research 219(2):425–433

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

Zanda S, Zuddas P, Seatzu C (2018) Long term nurse scheduling via a decision support system based on linear integer programming: A case study at the university hospital in cagliari. Computers and Industrial Engineering 126:337–347, DOI 10.1016/j.cie.2018.09.027