An integer programming approach for the physician rostering problem

Toni I. Wickert · Alberto F. Kummer Neto · Luciana S. Buriol

Abstract Nurse and physician rostering problems seek to find an optimal way to assign nurses and physicians to shifts respecting a set of hard and soft constraints. If a single hard constraint is violated, the solution is considered infeasible, while soft constraints violations are penalized in the objective function. This paper presents an integer programming model and a matheuristic algorithm for the physician rostering problem. Moreover, a comparison of the physician and nurse rostering is analyzed to identify common constraints present in both problems. The studied Nurse Rostering Problem (NRP) is based on the constraints proposed for the Second International Nurse Rostering Competition (INRC-II), while the Physician Rostering Problem (PRP) presents constraints provided by Hospital de Clínicas de Porto Alegre (HCPA), Brazil. Due to the difficulty of solving large instances, a matheuristic is proposed to tackle the problem. Results demonstrate that the matheuristic generated near-optimal results within an acceptable computational time limit.

1 Introduction

Healthcare personnel rostering is a common problem found in healthcare institutions and often a difficult task to be done manually. Due to a high number of possibilities, schedules organization is time-consuming and many times generates poor results. The

Toni I. Wickert
Institute of Informatics, Federal University of Rio Grande do Sul (UFRGS), Porto Alegre, Brazil and KU Leuven, Department of Computer Science, CODeS & imec, Gebroeders De Smetstraat 1, 9000 Gent, Belgium
E-mail: tiwickert@inf.ufrgs.br

Alberto Francisco Kummer Neto
Institute of Informatics, Federal University of Rio Grande do Sul (UFRGS), Porto Alegre, Brazil
E-mail: afkneto@inf.ufrgs.br

Luciana S. Buriol
Institute of Informatics, Federal University of Rio Grande do Sul (UFRGS), Porto Alegre, Brazil
E-mail: buriol@inf.ufrgs.br

automation of this task, using computational techniques, brings a series of benefits such as better employees distribution during the rosters, less time to organize the rosters and consequently an overall cost reduction.

In the literature, the most common problem found related to healthcare rostering is the Nurse Rostering Problem (NRP). There is a wide literature where the problem was approached in hospitals specific problems such as [5,6,12]. Moreover, two competitions were organized such that the researchers could test their solving methods and compare results among participants. These are the cases of the INRC-I [11] and INRC-II [7].

A similar problem is the physician rostering, that aims to generate rosters for physicians in a variety of areas. For example, physicians can be scheduled to hospitals emergencies, intensive care units, and cancer treatment units, according to the demand of each area and the physician's specialty. The Physician Rostering Problem (PRP) received less attention in the literature when compared to the NRP.

This work approaches the PRP with a specific study based on the data provided by Hospital de Clínicas de Porto Alegre (HCPA), Brazil. An integer programming formulation was proposed, and an extension to the INRC-II instances was developed to add some specific constraints for the PRP. Moreover, 10 instances with 50, 100 and 150 physicians were generated and approached using a MIP solver and a Fix-and-Optimize (F&O) matheuristic.

The present paper addresses three primary research questions, namely:

- Is it possible to solve small and medium size instances using integer programming and a MIP solver in a reasonable time?
- What are the differences between the constraints present in the INRC-II instances and a real-world PRP?
- Can a matheuristic developed for the NRP be adapted, with a few changes, and generate good results for the PRP?

The remaining paper is organized as follows. Section 2 presents the literature review. Section 3 presents the PRP constraints and a comparison with the constraints of the NRP proposed for the INRC-II. Section 4 presents the integer programming formulation for the PRP. Section 5 presents the F&O matheuristic employed to solve the PRP instances. Section 6 presents the results obtained by the F&O matheuristic and a comparison with a MIP solver, while Section 7 presents the conclusions and future work.

2 Literature Review

Personnel rostering has received considerable attention in the literature. The work of [9] presents an overview of models and methods concerning the general personnel rostering. Interested readers are referred to [8] for a general overview regarding PRP.

The work developed by [2] presents an Integer Programming (IP) model for the PRP and employes a MIP solver to solve the model. The IP model satisfies all service requirements and contractual agreements (including rest periods and annual leave), while trying to respect, as much as possible, employees preferences with particular attention on workload balancing, and additional preference aspects related to each physician. They applied the developed model in a real-world case of physician scheduling in some of the departments of one of the largest Italian university hospitals. Reported results demonstrate the effectiveness of the proposed optimization approach when compared to the solution obtained manually.

The approach proposed by [16] tackles a physician scheduling problem with flexible shifts. The problem consists of a fixed number of identical physicians who are scheduled over a planning horizon, where each day consists of periods of one hour. They solved the problem using a decomposition heuristic where the overall problem is decomposed into weekly subproblems. The decomposition heuristic generated the best results in less time when compared to the other solving methods, and a previous work [3].

The authors of [4] developed a column generation to tackle the physician scheduling problem with different experience levels in hospitals. The data were provided by an anesthesia department of an 1100-bed hospital. They employed the column generation procedure because the MIP solver was not able to solve the weekly subproblems up to optimality within several hours. The authors developed a column generation based heuristic to obtain integer solutions. Computational results show the efficiency of the proposed algorithm in finding near-optimal and optimal solutions.

The work of [10] studied a real-world problem in a surgery department of a large government hospital of Singapore. Instead of assigning physicians to shifts, physicians are assigned to a set of tasks incorporating a large number of constraints and complex physician preferences. For small and medium instances they employed a MIP solver, while to address the larger instances they developed a two-phase local search algorithm: Phase 1 consists of a greedy heuristic that is used to initialize a solution, while in phase 2 a local search algorithm with different types of neighborhood structures is used to improve the solution. The experiments demonstrate that the heuristic generates good results in an acceptable time limit when employing the larger instances.

A solving method using constraint programming combined with local search and some ideas from genetic algorithms was proposed by [14]. The constraint programming model was used to solve the hard constraints. After this process, a set (or population) of solutions are generated by starting the search on different days. Although this step usually takes only a few seconds, a cutoff time was imposed to avoid wasting time on difficult instances. To improve the pool of solutions, they combine attributes of the best solutions using a crossover operator. This combination generates a partial solution to the problem which is then completed by invoking the MIP solver for the incomplete part, a similar technique of large neighborhood search. This evolutionary process tends to generate better solutions at each iteration. The process terminates when the best solution is found or until a time limit. For the experiments, instances from two Canadian hospitals were employed.

A genetic algorithm to schedule physicians for emergency rooms was proposed by [13]. An encoding called doctor-shift view that assigns doctors to shifts was developed. The design of a heuristic-schedule allows to create an initial population of feasible solutions. Specific crossover operators were necessary to be implemented to allow the exchange of whole work weeks, together with a repair function. With this function, it is possible the creation of new populations with feasible solutions. The experiments were based on instances provided by a Spanish hospital. The proposed genetic algorithm was tested in different situations, generating good quality solutions as well as time-saving in designing the rosters.

A MIP formulation for the integrated physician and surgery scheduling problem was approached by [17]. The model is based on the most frequently observed objectives and restrictions of the surgery scheduling and the physician rostering problem in the literature. Data provided by a Belgian hospital was employed to validate the experiments. Experiments using a MIP solver were conducted employing instances up to 5 operating rooms, 50 surgeries and 20 physicians on a weekly schedule planning horizon.

The difference of this work compared to the existing literature is regarding the model and solving method. In this approach physicians work in multiple locations, so the solving method needs to decide which is the best working location for each day and shift during the planning horizon. Moreover, double shift assignments on weekends and holidays are considered to attend the hospital requirements.

3 Physician rostering problem constraints

According to our study, the set of constraints proposed for the INRC-II are the most similar compared to the tackled PRP. Table 1 presents a comparison between the constraints of the INRC-II and the data provided for the PRP. As can be observed, there are eight hard constraints, two are common in both problems, and six are specific from NRP or PRP. There are fourteen soft constraints, seven are present in both, and seven are specific for each problem.

The primary differences of this specific PRP, compared other PRP and NRP found in the literature, is that a physician must be assigned to both day shifts, or one Night shift, or none shift on weekends and holidays. The proposed integer programming formulation covers the following constraints:

Hard constraints

H1. A physician can be assigned to at most one shift per day during weekdays;
H2. A physician must be assigned to both day shifts, or one Night shift, or none shift on weekends and holidays;
H3. A shift type succession must belong to a valid succession (for example, a Night shift cannot be followed by an Early or Late shift);
H4. A physician can be available at only some (or none if in vacation) shifts to work;
H5. The worked shifts must be at the same location in a single day on Saturdays, Sundays and holidays. For example, if a physician works on Saturday during Early and Late shift, both shifts must be at same location;

Soft Constraints

S1. Minimum number of physician per day/shift/location;
S2. Maximum number of physician per day/shift/location;
S3. Maximum number of consecutive assignments per Night shifts;

Table 1 Comparison between the constraints proposed by the INRC-II and the real world PRP

Hard Constraints	**INRC II**	**PRP**
A nurse/physician can be assigned to at most one shift per day	X	-
A nurse/physician can be assigned to at most one shift per day on weekdays	X	X
A nurse/physician must be assigned to both day shifts, or a Night shift, or have a day off on weekends and holidays	-	X
Minimum number of nurses/physicians by day/shift/skill	X	-
A shift type succession must belong to a valid succession (ex.: Night shift followed by Early shift is not allowed)	X	X
A shift of a given skill must necessarily be fulfilled by a nurse/physician having that skill	X	-
A nurse/physician can be available at only some (or none if in vacation) shifts to work	-	X
On the day shifts on weekends and holidays, the nurse/physician must work at same location	-	X
Soft Constraints	**INRC II**	**PRP**
Preferred number of nurses/physicians by day/shift/skill	X	-
Minimum number of nurses/physician by day/shift/location	-	X
Maximum number of nurses/physician by day/shift/location	-	X
Minimum number of consecutive assignments by shift	X	-
Maximum number of consecutive assignments by shift	X	X
Minimum number of consecutive assignments	X	-
Maximum number of consecutive assignments	X	X
Minimum number of consecutive days off	X	-
Maximum number of consecutive days off	X	-
An undesired working day/shift is penalized	X	X
It is preferred that a nurse/physician works on the two days of a weekend, or none	X	X
Minimum number of working days by schedule	X	X
Maximum number of working days by schedule	X	X
Maximum number of working weekends	X	X

S4. Maximum number of consecutive assignments;
S5. An undesired working day/shift is penalized;
S6. It is preferred that a physician works on the two days of a weekend, or none;
S7. Minimum number of working days over the planning horizon;
S8. Maximum number of working days over the planning horizon;
S9. Maximum number of working weekends.

A summary of the input data files generated to solve the physician rostering is provided below. The input format is based on the INRC-II files, with a few changes to attend the differences between the constraints of each problem.

– *Scenario:* contains the planning horizon (number of weeks), locations (in-patient unit 1, in-patient unit 2, emergency room 1, emergency room 2), contracts (full-time, part-time), physician identifications (names), shift types (Early, Late, Night);
– *Week Data:* minimum and maximum number of physicians for each day/shift/location, days/shifts-off requests, days/shifts that a physician is not available;

- *History:* border data to check the constraints regarding consecutive assignments (for example, maximum consecutive working days on same shift and maximum consecutive working days).

3.1 Example of a physician rostering problem

Table 2 represents a simple roster presented in the physician-day view, containing three physicians (Physician1, Physician2 and Physician3), three shifts (Early [E], Late [L] and Night [N]), and three locations (In-patient Unit1 [1], In-patient Unit2 [2], and In-patient Unit3 [3]). The day shifts (Early and Late) have 6 hours, while the Night shifts have 12 hours. The shifts are organized as follows:

- Early (6h): 08-14h;
- Late (6h): 14-20h;
- Night (12h): 20-08h.

The basic unit is a 6h-shift, once a physician is not allowed to work partial shifts. The Night shift has 12h, while the day shifts (Early and Late) have 6h. By this way, if a physician works a Night shift it is considered as two worked shifts. This procedure is necessary to calculate the total number of working shifts during the planning horizon.

As an example, Physician1 works on Monday on Late shift at location In-patient Unit1, on Tuesday on Late shift at location In-patient Unit2, and on Saturday/Sunday on both day shifts (Early and Late) at location In-patient Unit1. Dashes represent a day off. Marked in gray, are the day shifts (Early and Late) on Saturday and Sunday, that must be worked together, and the Nights shifts, with 12 working hours, which need to be considered twice to calculate the total number of worked shifts per roster.

Table 2 Example of a roster with seven days and three physicians.

Physician	Mon	Tue	Wed	Thu	Fri	Sat	Sun
Physician1	L[1]	L[2]	N[1]	–	–	E/L[1]	E/L[1]
Physician2	N[2]	N[3]	–	–	–	N[3]	N[2]
Physician3	–	E[1]	–	L[2]	N[1]	–	–

4 Integer Programming Formulation

In this section, the proposed integer programming formulation is presented, considering the hard and soft constraints for the PRP. Several IP formulations can be found in the literature regarding staff rostering. Interested readers are referred to [1] and [15] for a general overview of common constraints for personnel rostering and complexity analysis.

Table 3 presents the indices (first column) used to identify variables associated with the constraint on the formulation. The second column describes the constraints, the third column (Id) is the identifier, and the last column presents the weight for the violation of the constraint. For example, in the objective function in Eq. 1 the first term $a^1_{dsk}\omega^1$ refers to the variables related to the minimum number of physicians violation (identified by index 1 in Table 3) multiplied by its respective weight ω^1 (10000). The other constraints follow the same idea.

Table 3 Soft constraints indices used to associate each constraint with its respective variable in the formulation. For each index is presented a description and its weight.

Index	Constraint Description	Id	Weight
1	Minimum number of physicians	S1	10000
2	Maximum number of physicians	S2	10000
3	Maximum consecutive assignments to the same shift	S3	15
4	Maximum consecutive assignments (worked days)	S4	30
5	Physician undesired working day/shift	S5	10
6	Complete weekend	S6	30
7	Minimum number of assignments over the planning horizon	S7	20
8	Maximum number of assignments over the planning horizon	S8	20
9	Maximum number of working weekends	S9	30

Table 4 presents the sets, decision and auxiliary variables employed in the formulation. The objective function minimizes the cost associated with the violation of the soft constraints. Equations 2 to 22 are the constraints.

Table 4: Indices, sets and variables used in the mathematical formulation.

Symbol	Definition
Input Data	
$n \in N$	n is the index of the physician, and N is the set of physicians;
$d \in D$	d is the index of the day, and D is the set of days;
$d \in \tilde{D}$	d is the index of the weekend day or holiday, and $\tilde{D}$ is the set of weekend days and holidays;
$s \in S$	s is the index of the shift, and $\{1,2,3\} \in S$ is the set of shifts, where 1 corresponds to Early, 2 to Late and 3 to Night;
$k \in K$	k is the index of the location, and $\{1,2,3\} \in K$ is the set of locations, where 1 corresponds to Emergency1, 2 to Emergency2 and 3 to Emergency3;
$l_{nk} \in \{0,1\}$	l_{nk} is the index of the allowed location k for physician n. Considering the case in which *Physician1* is allowed to work on *Emergency1* and *Emergency2*, and the problem's input has three locations. The vector for *physician1* is $l_{nk} = \{1,1,0\}$ and represents that the *Physician1* can work at location *Emergency1* and *Emergency2* and is not allowed to work at *Emergency3*.
$(n,d,s) \in R$	set containing triples with the day d, shift s for physician n that they are not available to work. $R = \{n,d,s\}$, for example, $R = \{1,2,2\}$ means *Physician1* is not available to work on day 2, shift 2 (Late shift).

$(n,d,s) \in U$	set containing triples with the undesired working day d, shift s for physician n. $U = \{n,d,s\}$, for example, $U = \{1,2,3\}$ means *Physician1* prefers to avoid working on day 2, shift 3 (Night shift).
$(s',s'') \in \hat{S}$	set containing the pairs of invalid shift successions $\hat{S} = \{s',s''\}$, for example, $\hat{S} = \{3,1\}$ means that a Night shift cannot be followed by an Early shift;
$w \in W$	w is a Saturday index and W the set of all Saturdays indices;
α^i_{dsk}	limit of soft constraint $i \in 1,2$, that is, minimum and maximum number of physicians per day d, shift s, location k;
β^i_n	limit of soft constraint $3,4,7,8,9$, that is, maximum consecutive assignment to the same shift, maximum consecutive working days, minimum/maximum number of assignments over the schedule period, and maximum working weekends, for physician n;
ω^i_n	weight for violating the lower and upper limits of soft constraint i for nurse n.

Decision Variables

$x_{ndsk} \in \{0,1\}$	1 if physician n is allocated to shift s, day d, and location k, 0 otherwise;
$y_{nw} \in \{0,1\}$	1 if physician n works at weekend w, 0 otherwise.
$z_{nd} \in \{0,1\}$	1 if physician n works on day d, during both day shifts, 0 otherwise.
$o_{nd} \in \{0,1\}$	1 if physician n is allocated to work on day d, 0 otherwise.

Auxiliary Variables

$a^i_{dsk} \in \mathbb{N}^*$	number of violations of the soft constraint $i \in (1,2)$ for day d, shift s, location k;
$c^i_{nd} \in \mathbb{N}^*$	number of violations of the soft constraint $i \in (3,4)$ for physician n on day d;
$g^5_{nds} \in \mathbb{N}^*$	number of violations of the soft constraint 5 for physician n on day d, shift s;
$h^6_{nw} \in \mathbb{N}^*$	number of violations of the soft constraint 6 for physician n on weekend w;
$j^i_n \in \mathbb{N}^*$	number of violations of the soft constraint $i \in (7..9)$ for physician n.

Before the solving method starts, a file is read containing the border data from the previous month, that is, the total number of assignments, last assigned shift type, number of consecutive assignments of the last shift type, and number of consecutive worked days. These data are necessary to calculate the constraints: minimum/maximum number of assignments, maximum number of working weekends, invalid shift type succession (hard constraint), maximum number of consecutive assignments on the same shift, and maximum number of consecutive working days, respectively.

$$
\begin{aligned}
\textbf{Min}\quad & \left[\sum_{d\in D}\sum_{s\in S}\sum_{k\in K}\sum_{i=1,2} a^i_{dsk}\omega^i\right] + \left[\sum_{n\in N}\sum_{d\in D}\sum_{i=3,4} c^i_{nd}\omega^i\right] + \\
& \left[\sum_{n\in N}\sum_{d\in D}\sum_{s\in S} g^5_{nds}\omega^5\right] + \left[\sum_{n\in N}\sum_{w\in W} h^6_{nw}\omega^6\right] + \\
& \left[\sum_{n\in N}\sum_{i=7..9} j^i_n\omega^i\right]
\end{aligned} \tag{1}
$$

Subject to

$$\sum_{s\in S}\sum_{k\in K} x_{ndsk} \le 1 \qquad \forall n\in N, d\in D\setminus\tilde{D} \tag{2}$$

$$\sum_{k\in K}(x_{n\tilde{d}sk}+x_{n\tilde{d}(s+1)k}) = 2z_{n\tilde{d}} \qquad \forall n\in N, \tilde{d}\in\tilde{D}, s=1 \tag{3}$$

$$\sum_{k\in K} x_{ndsk} + z_{nd} \le 1 \qquad \forall n\in N, d\in\tilde{D}, s=3 \tag{4}$$

$$\sum_{s\in S}\sum_{k\in K} x_{ndsk} \le 2o_{nd} \qquad \forall n\in N, d\in D \tag{5}$$

$$\sum_{k\in K}(x_{nds'k}+x_{n(d+1)s''k}) \le 1 \qquad \forall n\in N, d\in D\setminus\{|D|\}, (s',s'')\in\hat{S} \tag{6}$$

$$(1-l_{nk})x_{ndsk} = 0 \qquad \forall n\in N, d\in D, s\in S, k\in K \tag{7}$$

$$\sum_{k\in K} x_{ndsk} = 0 \qquad \forall (n,d,s)\in R \tag{8}$$

$$x_{ndsk} - x_{nd(s+1)k} = 0 \qquad \forall n\in N, d\in\tilde{D}, s=1, k\in K \tag{9}$$

$$\sum_{n\in N} x_{ndsk} + a^1_{dsk} \ge \alpha^1_{dsk} \qquad \forall d\in D, s\in S, k\in K \tag{10}$$

$$\sum_{n\in N} x_{ndsk} - a^2_{dsk} \le \alpha^2_{dsk} \qquad \forall d\in D, s\in S, k\in K \tag{11}$$

$$\sum_{d'=d}^{\beta^3_n+d}\sum_{k\in K} x_{nd'sk} - c^3_{nd} \le \beta^3_n \qquad \forall n\in N, d\in\{1,\dots,|D|-\beta^3_n\}, s=3 \tag{12}$$

$$\sum_{d'=d}^{\beta^4_n+d} o_{nd'} - c^4_{nd} \le \beta^4_n \qquad \forall n\in N, d\in\{1,\dots,|D|-\beta^4_n\} \tag{13}$$

$$\sum_{k\in K} x_{ndsk} \le g^5_{nds} \qquad \forall (n,d,s)\in U \tag{14}$$

$$o_{nw} + o_{n(w+1)} + h^6_{nw} = 2y_{nw} \qquad \forall n\in N, w\in W \tag{15}$$

$$\begin{aligned}&\sum_{d\in D}\sum_{s\in\{1,2\}}\sum_{k\in K} x_{ndsk} + \\ &\sum_{d\in D}\sum_{s=3}\sum_{k\in K} 2x_{ndsk} + j^{10}_n \ge \beta^7_n \qquad \forall n\in N\end{aligned} \tag{16}$$

$$\begin{aligned}&\sum_{d\in D}\sum_{s\in\{1,2\}}\sum_{k\in K} x_{ndsk} + \\ &\sum_{d\in D}\sum_{s=3}\sum_{k\in K} 2x_{ndsk} - j^{11}_n \le \beta^8_n \qquad \forall n\in N\end{aligned} \tag{17}$$

$$\sum_{w\in W} y_{nw} - j^9_n \le \beta^9_n \qquad \forall n\in N \tag{18}$$

$$x_{ndsk}\in\{0,1\} \qquad \forall n\in N, d\in D, s\in S, k\in K \tag{19}$$

$$y_{nw} \in \{0,1\} \qquad \forall n \in N, w \in W \qquad (20)$$
$$z_{nd} \in \{0,1\} \qquad \forall n \in N, d \in D \qquad (21)$$
$$o_{nd} \in \{0,1\} \qquad \forall n \in N, d \in D \qquad (22)$$

Constraints 2-9 are the hard constraints. Constraints 2 ensure a physician can be assigned to at most one shift per day during weekdays. Constraints 3 and 4 ensure that a physician must be assigned to both day shifts, or one Night shift, or none shift on weekends and holidays. Constraints 5 stores in a auxiliary variable o_{nd} if physician n works on day d. Constraints 6 ensure a shift type succession must belong to a valid succession (for example, a Night shift cannot be followed by an Early shift). Constraints 7 ensure a physician must be allowed to work in the required location. Constraints 8 ensure a physician is scheduled only if they are available. Constraints 9 ensure a physician that is working on a day shift on weekends and holidays work both shifts on the same location.

Constraints 10 calculate the minimum number of physicians violations. Constraints 11 calculate the maximum number of physicians violations. Constraints 12 calculate the maximum number of consecutive assignments to Night shifts violations. Constraints 13 calculate the maximum consecutive assignments (worked days) violations. Constraints 14 calculate the undesired worked day or shift violations. Constraints 15 calculate the complete weekend violations. Constraints 16 calculate the minimum number of assignments over the scheduling period violations. Constraints 17 calculate the maximum number of assignments over the scheduling period violations. Constraints 18 calculate the maximum number of working weekends violations. Constraints 19-22 define the decision variables as binary.

5 Fix-and-optimize matheuristic for the PRP

The use of hybrid methods, which combine mathematical programming with heuristics, such as the F&O matheuristics, has grown in the last years. The proposed algorithm was adapted from a previous version employed to address the NRP [18]. In this section, the F&O matheuristic developed to solve the PRP is detailed.

First, a feasible solution is generated employing a solver and only considering the hard constraints. Then, iteratively a subset of variables are fixed to their current values, and the solver is employed to optimize the resulting subproblem. All hard and soft constraints are considered when the subproblem is solved.

Algorithm 1 receives several parameters where *kMax {Day, Physician, Week, Shift}* represent the maximum number of free variables of each type and *kLimit {Day, Physician, Week, Shift}* are the limits of permutations generated by each type of neighborhood. For instance, considering that there are 5 physicians and *kPhysician=1*, it is possible to generate 5 types of free physicians to optimize: *PhysicianFreePermutationSet([1], [2], [3], [4], [5])*. However, if *kPhysician=2* there are 10 possibilities: *PhysicianFreePermutationSet([1,2], [1,3], [1,4], [1,5], [2,3], [2,4], [2,5], [3,4], [3,5], [4,5])*. If *kLimitPhysician=5*, only 5 items will be randomly added to the set *PhysicianFreePermutationSet*. If the *kLimitPhysician* $\geq$ *10* all the possible permutations will be added to the set *PhysicianFreePermutationSet*.

Algorithm 1: Fix-and-optimize matheuristic algorithm.

```
1  FixAndOptimize(kMaxWeek, kLimitWeek, kMaxShift, kLimitShift, kMaxDay, kLimitDay, kMaxPhysician,
     kLimitPhysician, TL, STL);
2  x = generateInitialSolution();
3  kWeek = kShift = kDay = kPhysician = 1;
4  do
5      x = fixByDay(x, kDay, kLimitDay, STL);
6      kDay = kDay+1;
7      x = fixByPhysician(x, kPhysician, kLimitPhysician, STL);
8      kPhysician=kPhysician+1;
9      x = fixByWeek(x, kWeek, kLimitWeek, STL);
10     kWeek=kWeek+1;
11     x = fixByShift(x, kShift, kLimitShift, STL);
12     kShift=kShift+1;
13     if (kDay > kMaxDay) kDay = 1;
14     if (kPhysician > kMaxPhysician) kPhysician = 1;
15     if (kWeek > kMaxWeek) kWeek = 1;
16     if (kShift > kMaxShift) kShift = 1;
17 while TL;
18 return x;
```

Algorithm 1 starts generating an initial feasible solution *x* (line 2) considering only the hard constraints using a MIP solver. The number of free variables to optimize is initialized with one (line 3), and the loop (lines 4 to 17) is iterated until the time limit (TL) is reached.

Inside the loop (lines 5 to 16), the algorithm calls different neighborhoods. Each neighborhood is explored until finding a local minimum, or it reaches the STL (subproblem time limit). For each step, the value of *k{Day, Physician, Week, Shift}* is increased by 1. If the limit of each type is exceeded the variables are reset to 1 (lines 13 to 16).

Algorithm 2 starts generating the permutations of the physicians that will be free to be optimized, until the *kLimitPhysician* is reached (function *permutation* at line 2). The loop (lines 4 to 16) is iterated until no improvement of 20% is found. The loop starts storing the current solution value and the best neighbor value (function *OFV* lines 5 and 6). The nested loop (lines 7 to 14) explores the neighborhood fixing the entire problem (line 8), and unfixing only the free variables that will be optimized (line 9). The MIP solver is called and executed until the optimal solution is found or STL is reached (line 10). If the Objective Function Value (OFV) of the subproblem *x* is lower than the OFV of the best neighbor (line 11) the *bestNeighborValue* variable is updated (line 12).

Tables 5 and 6 detail an iteration of a *fix by physician* neighborhood with *kPhysician = 1* and *kPhysician = 2*, respectively. The rows (Physicians) in gray are free to be optimized by the solver. Since the *fix by week*, *fix by shift* and *fix by day* follow the same idea, the pseudo-code of these algorithms were omitted.

6 Computational results

The source-code was written in Java and compiled with OpenJDK 1.8. The experiments were conducted on an Intel Core i5-2410M CPU @ 2.30GHz x2 with 6GB of RAM memory running Linux Mint 17.2 64-bits. The solver employed was CPLEX

Algorithm 2: Fix by physician.

```
FixByPhysician(x, kPhysician, kLimitPhysician, STL);
physicianFreePermutationSet[] = permutation(kPhysician, kLimitPhysician);
improved = false;
do
    currentSolutionValue = OFV(x);
    bestNeighborValue = OFV(x);
    foreach Integer[] free : physicianFreePermutationSet do
        fixAll(x);
        unFix(free, x);
        solve(x, STL);
        if OFV(x) < bestNeighborValue then
            bestNeighborValue = OFV(x);
        end
    end
    improved = bestNeighborValue*1.2 < currentSolutionValue;
while improved;
return x;
```

Table 5 Fix by physician (kPhysician=1).

Physician	Mon	Tue	Wed
P1	L[1]	L[2]	N[2]
P2	N[1]	N[1]	N[3]
P3	–	E[3]	–

↓

P1	L[1]	N[2]	N[2]
P2	N[1]	N[1]	N[3]
P3	–	E[3]	–

↓

P1	L[1]	N[2]	N[2]
P2	N[2]	N[1]	N[3]
P3	–	E[3]	–

Table 6 Fix by physician (kPhysician=2).

Physician	Mon	Tue	Wed
P1	L[1]	N[2]	N[2]
P2	N[2]	N[1]	N[3]
P3	–	E[3]	–

↓

P1	L[1]	N[1]	N[2]
P2	N[2]	L[1]	N[3]
P3	–	E[3]	–

↓

N1	L[1]	N[1]	N[2]
N2	E[2]	L[1]	N[3]
N3	E[3]	E[3]	–

version 12.6.2. The default CPLEX parameters were used. The *gap* (last column of Table 7) is calculated using the equation $gap = 100 \times (MIP - F\&O)/min(MIP, F\&O)$, MIP and F&O mean the objective function value (OFV) of the fifth and seventh columns in Table 7.

The dataset employed in the experiments were generated based on the data provided by HCPA. The algorithm was tested in 30 generated instances. Currently, the real number of physicians to schedule is 50. However, the number of physicians will increase in the future, that is the reason for generating larger instances. The objective is to analyze whether they still can be solved using the proposed methods. The following instances were generated, namely:

- 10 instances with 50 physicians and 4 weeks;
- 10 instances with 100 physicians and 4 weeks;
- 10 instances with 150 physicians and 4 weeks.

The time limit for each experiment was fixed according to the instance size and algorithm:

- One single long run using the MIP solver with time limit of 12h;
- 10 runs using the MIP solver with different seeds, considering time limit of 20, 40 and 60 minutes for the instances with 50, 100 and 150 physicians, respectively;
- 10 runs using F&O Matheuristic with time limit of 10, 20 and 30 minutes for the instances with 50, 100 and 150 physicians, respectively;

Table 7 details the results. The first column presents the Instance Id, the second column presents the MIP solver LB (lower bound) within a time limit of 12h, the third column presents the OFV obtained by the MIP solver, and the fourth column presents the relative GAP provided by the MIP solver.

Table 7 Results for PRP instances with 50, 100 and 150 physicians.

Instance	MIP LB	Single Run MIP 12h	MIP Gap(%)	AVG 10 runs MIP	Std. Dev.	AVG 10 runs F&O	Std. Dev.	MIP vs F&O Gap(%)
p050_inst_01	30,305	30,305	0.00	30,305	0	30,365	8	0.20
p050_inst_02	30,460	30,460	0.00	30,460	0	30,500	10	0.13
p050_inst_03	30,505	30,505	0.00	30,505	0	30,575	25	0.23
p050_inst_04	30,965	30,965	0.00	30,965	0	30,985	14	0.06
p050_inst_05	30,685	30,685	0.00	30,685	0	30,695	18	0.03
p050_inst_06	31,705	31,705	0.00	31,705	0	31,755	18	0.16
p050_inst_07	30,015	30,015	0.00	30,015	0	30,025	24	0.03
p050_inst_08	30,215	30,215	0.00	30,215	0	30,275	24	0.20
p050_inst_09	31,670	31,670	0.00	31,670	0	31,670	20	0.00
p050_inst_10	30,765	30,765	0.00	30,765	0	30,805	29	0.13
average			0.00					0.12
p100_inst_01	24,429	25,525	4.29	26,320	8,678	25,845	57	-1.84
p100_inst_02	26,720	27,945	4.38	29,940	102,042	28,435	58	-5.29
p100_inst_03	25,082	26,300	4.63	175,395	115,256	26,650	68	-558.14
p100_inst_04	24,280	25,285	3.97	66,040	150,074	25,515	75	-158.83
p100_inst_05	24,633	25,775	4.43	28,615	16,339	26,060	72	-9.80
p100_inst_06	25,660	26,920	4.68	28,960	12,819	27,130	51	-6.75
p100_inst_07	23,205	24,505	5.31	26,075	971	24,870	84	-4.85
p100_inst_08	25,282	26,445	4.40	97,175	103,925	26,770	90	-263.00
p100_inst_09	25,946	27,130	4.36	48,620	90,774	27,560	76	-76.42
p100_inst_10	23,775	25,030	5.01	28,185	43,730	25,430	135	-10.83
average			4.55					-109.57
p150_inst_01	55,742	60,030	7.14	33,866,360	12,053,710	61,425	158	-55,034.49
p150_inst_02	54,011	58,320	7.39	28,101,150	12,667,878	60,110	202	-46,649.54
p150_inst_03	54,135	58,070	6.78	5,985,325	12,871,189	59,625	228	-9,938.28
p150_inst_04	53,457	57,595	7.18	28,062,400	12,378,134	59,090	235	-47,390.95
p150_inst_05	54,786	59,740	8.29	33,867,280	13,946,338	90,540	202	-37,305.88
p150_inst_06	54,170	58,720	7.75	5,563,180	10,155,870	69,920	238	-7,856.49
p150_inst_07	53,959	57,570	6.27	12,057,155	10,809,383	59,255	282	-20,247.91
p150_inst_08	53,199	56,750	6.26	7,150,135	10,581,572	59,120	218	-11,994.27
p150_inst_09	53,396	57,580	7.27	33,868,180	11,909,774	57,085	374	-59,229.39
p150_inst_10	51,782	55,810	7.22	7,480,425	11,897,493	57,148	245	-12,989.68
average			7.15					-30,863.69

The fifth column presents the average OFV obtained by the MIP solver when running with the time limit of 20, 40 and 60 minutes for the instances with 50, 100 and 150 physicians, and the sixth column the respective standard deviation. The seventh column presents the average OFV obtained by the F&O matheuristic, and the eighth column the respective standard deviation. The F&O matheuristic time limit was 10, 20 and 30 minutes for the instances with 50, 100 and 150 physicians, respectively. The last column presents the relative gaps of the results obtained by 10 runs employing the MIP solver compared to 10 runs using the F&O matheuristic.

As can be observed, at the third and fourth columns, within a limit of 12h, the MIP solver solved to the optimality all instances with 50 physicians. The instances with 100 and 150 physicians were solved near-optimal, with average relative gaps of 4.55% and 7.15%, respectively.

When the time limit of the MIP solver is decreased (fifth and sixth columns), only the instances with 50 physicians were solved to optimality. The instances with 100 and 150 physicians have a huge standard deviation, and the MIP solver was unable to generate good results employing the proposed IP formulation. The results of the F&O matheuristic (seventh and eight columns) detail that even with the half of the time compared to the MIP solver, the algorithm generated good results. Near-optimal results were obtained employing the instances with 50 physicians and better results compared to the MIP solver using the instances with 100 and 150 physicians.

7 Conclusions

This paper presented an integer programming formulation and a F&O matheuristic for the PRP. Moreover, a comparison between the constraints present in the INRC-II instances and the studied physician rostering demonstrate the similarities and differences between the problems.

The computational experiments demonstrate that the MIP solver could solve the small instances with 50 physicians to the optimality. For larger instances, with 100 and 150 physicians, the MIP solver only generated good results when the time limit was increased to 12h, using the proposed IP formulation.

The proposed F&O matheuristic generated good results within short time limits. Results near-optimal were generated for instances with 50 physicians in 10 minutes. Moreover, employing instances with 100 and 150 physicians, even with acceptable computational time limits (20 and 30 minutes), the F&O matheuristic generated good results.

The primary contributions of this work are an integer programming formulation for a real-world PRP, extensions to the INRC-II instances to attend specific PRP constraints and a F&O matheuristic that generated good results within short time limits. Logs of the executions, instances, and results are available on-line[1].

Future research will consider the implementation of the proposed integer programming formulation and the F&O matheuristic using an open-source solver.

Acknowledgements The authors would like to thank staff members of Hospital de Clínicas de Porto Alegre (HCPA) that collaborate with this work. The second author would like to thank the CAPES–Coordination for the Improvement of Higher Education Personnel–for his doctoral scholarship. This research has the support of FAPERGS, project PqG 17/2551-0001201-1.

[1] `http://www.inf.ufrgs.br/~tiwickert/download/2017/physician`

References

1. Brucker, P., Qu, R., Burke, E.: Personnel scheduling: Models and complexity. European Journal of Operational Research **210**(3), 467 – 473 (2011). DOI 10.1016/j.ejor.2010.11.017
2. Bruni, R., Detti, P.: A flexible discrete optimization approach to the physician scheduling problem. Operations Research for Health Care **3**(4), 191 – 199 (2014). DOI http://dx.doi.org/10.1016/j.orhc.2014.08.003
3. Brunner, J.O., Bard, J.F., Kolisch, R.: Flexible shift scheduling of physicians. Health Care Management Science **12**(3), 285–305 (2009). DOI 10.1007/s10729-008-9095-2. URL `http://dx.doi.org/10.1007/s10729-008-9095-2`
4. Brunner, J.O., Edenharter, G.M.: Long term staff scheduling of physicians with different experience levels in hospitals using column generation. Health Care Management Science **14**(2), 189–202 (2011). DOI 10.1007/s10729-011-9155-x. URL `http://dx.doi.org/10.1007/s10729-011-9155-x`
5. Burke, E.K., De Causmaecker, P., Petrovic, S., Vanden Berghe, G.: Metaheuristics for handling time interval coverage constraints in nurse scheduling. Applied Artificial Intelligence **20**(9), 743–766 (2006)
6. Burke, E.K., Li, J., Qu, R.: 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 (2010). DOI http://dx.doi.org/10.1016/j.ejor.2009.07.036
7. Ceschia, S., Dang, N.T.T., De Causmaecker, P., Haspeslagh, S., Schaerf, A.: Second international nurse rostering competition (2014). URL `http://mobiz.vives.be/inrc2/wp-content/uploads/2014/10/INRC2.pdf`
8. Erhard, M., Schoenfelder, J., Fügener, A., Brunner, J.O.: State of the art in physician scheduling (2016)
9. Ernst, A., Jiang, H., Krishnamoorthy, M., Sier, D.: Staff scheduling and rostering: A review of applications, methods and models. European Journal of Operational Research **153**(1), 3–27 (2004)
10. Gunawan, A., Lau, H.C.: Master physician scheduling problem. Journal of the Operational Research Society **64**(3), 410–425 (2013). DOI 10.1057/jors.2012.48. URL `http://dx.doi.org/10.1057/jors.2012.48`
11. Haspeslagh, S., De Causmaecker, P., Stølevik, M., Schaerf, A.: First international nurse rostering competition (2010). URL `https://www.kuleuven-kulak.be/~u0041139/nrpcompetition/nrpcompetition_description.pdf`
12. Petrovic, S., Vanden Berghe, G.: A comparison of two approaches to nurse rostering problems. Annals of Operations Research **194**(1), 365–384 (2012). DOI 10.1007/s10479-010-0808-9. URL `http://dx.doi.org/10.1007/s10479-010-0808-9`
13. Puente, J., Gómez, A., Fernández, I., Priore, P.: Medical doctor rostering problem in a hospital emergency department by means of genetic algorithms. Computers & Industrial Engineering **56**(4), 1232 – 1242 (2009). DOI http://doi.org/10.1016/j.cie.2008.07.016. URL `http://www.sciencedirect.com/science/article/pii/S0360835208001460`
14. Rousseau, L.M., Pesant, G., Gendreau, M.: A general approach to the physician rostering problem. Annals of Operations Research **115**(1), 193–205 (2002). DOI 10.1023/A:1021153305410. URL `http://dx.doi.org/10.1023/A:1021153305410`
15. Smet, P., Brucker, P., Causmaecker, P.D., Berghe, G.V.: Polynomially solvable personnel rostering problems. European Journal of Operational Research **249**(1), 67 – 75 (2016). DOI 10.1016/j.ejor.2015.08.025
16. Stolletz, R., Brunner, J.O.: Fair optimization of fortnightly physician schedules with flexible shifts. European Journal of Operational Research **219**(3), 622 – 629 (2012). DOI http://dx.doi.org/10.1016/j.ejor.2011.10.038. URL `http://www.sciencedirect.com/science/article/pii/S0377221711009787`. Feature Clusters
17. Van Huele, C., Vanhoucke, M.: Analysis of the integration of the physician rostering problem and the surgery scheduling problem. Journal of Medical Systems **38**(6), 43 (2014). DOI 10.1007/s10916-014-0043-z. URL `http://dx.doi.org/10.1007/s10916-014-0043-z`
18. Wickert, T.I., Sartori, C., Buriol, L.S.: A fix-and-optimize vns algorithm applied to the nurse rostering problem (2016). URL `http://iridia.ulb.ac.be/IridiaTrSeries/link/IridiaTr2016-007.pdf`