UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

VICTÓRIA SIMONETTI PORTELLA

# Mathematical Models and a Late Acceptance Fix-and-Optimize Approach for a Nurse Rostering Problem

Thesis presented in partial fulfillment
of the requirements for the degree of
Master of Computer Science

Advisor: Prof. PhD. Luciana Salete Buriol

Porto Alegre
August 2021

**CIP — CATALOGING-IN-PUBLICATION**

Portella, Victória Simonetti

Mathematical Models and a Late Acceptance Fix-and-Optimize Approach for a Nurse Rostering Problem / Victória Simonetti Portella. – Porto Alegre: PPGC da UFRGS, 2021.

55 f.: il.

Thesis (Master) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2021. Advisor: Luciana Salete Buriol.

1. INRC-II. 2. Integer programming. 3. Nurse rostering problem. 4. Matheuristics. 5. Fix-and-optimize. I. Buriol, Luciana Salete. II. Título.

*Math, it's just there. You're either right or you're wrong. That's what I like about it.*

— Katherine Johnson

## ACKNOWLEDGEMENTS

First and foremost, I would like to thank my advisor Luciana Buriol for her friendship, guidance, and unconditional support of my research. Thanks for her words of motivation and inspiration, especially while I was waiting for my baby.

Thanks to everyone in the Informatics Institute at the Federal University of Rio Grande do Sul (UFRGS) for the opportunities and high-quality education provided to me and my colleagues. Also, I would like to thank the National Council of Technological and Scientific Development (CNPq) for supporting my research.

I would like to thank all of my labmates at Optimization Research Group, especially Alberto. Furthermore, I am grateful to Olinto, who provided the computer facilities that I needed to complete my work.

I would like to thank my family, who always encouraged me to study. Also, I would like to thank all my friends, especially Isadora and Amanda, for their companionship and joy shared during this journey.

Finally, I extend the most generous gratitude to Árton and my little Benhur for all the love and unconditional patience.

## ABSTRACT

The Nurse Rostering Problem (NRP) is a classic optimization problem that has been extensively studied due to its practical and theoretical importance. It consists of assigning a given set of nurses to work shifts distributed into a planning horizon of multiple weeks. In addition to building a feasible schedule, a solution for the NRP needs to consider several requirements, such as skill-tailored assignments, labor laws, institutional norms and employee preferences. The high number of requirements associated with the combinatorial nature of the problem results in a process that could take several days to solve manually, even to produce a low-quality schedule. This work studies the static problem defined in the Second International Nurse Rostering Competition (INRC-II). Several techniques have been developed in the scientific literature to tackle the INRC-II problem, however, some issues are not explored yet. In this research, we investigate the limitations of MIP approaches, including how different requirements impact the resolution of the problem and how this approach could perform when inserted in a matheuristic procedure.

**Keywords:** INRC-II. integer programming. nurse rostering problem. matheuristics. fix-and-optimize.

**Modelos Matemáticos e uma *Matheuristic Late Acceptance Fix-and-Otimize* para um problema de escalonamento de enfermeiros**

## RESUMO

O Problema de Escalonamento de Enfermeiros (PEE) é um problema clássico de otimização que tem sido extensivamente estudado devido à sua importância prática e teórica. O PEE consiste em alocar um determinado conjunto de enfermeiros em turnos de trabalho distribuídos em um horizonte de planejamento de várias semanas. Uma escala de trabalho viável para o PEE precisa considerar diversos requisitos como leis trabalhistas, normas institucionais e preferências dos funcionários. O alto número de requisitos associados à natureza combinatória do problema resulta em um processo que pode levar vários dias para ser resolvido manualmente, e ainda produzir uma escala de baixa qualidade. Este trabalho estuda o problema estático definido na *Second International Nurse Rostering Competition* (INRC-II). Diversas técnicas têm sido desenvolvidas na literatura científica com objetivo de resolver o problema da INRC-II, entretanto, algumas questões ainda não foram exploradas. Nesta pesquisa, investigamos as limitações das abordagens MIP, incluindo como diferentes requisitos impactam na resolução do problema e como essa abordagem poderia funcionar quando inserida em um procedimento matemático.

**Palavras-chave:** INRC-II, Programação inteira, Problema de escalonamento de enfermeiros, *Matheuristics*, *fix-and-optimize*.

## LIST OF ABBREVIATIONS AND ACRONYMS

| | |
|---|---|
| INRC-I | First International Nurse Rostering Competition |
| INRC-II | Second International Nurse Rostering Competition |
| MIP | Mixed Integer Programming |
| MILP | Mixed Integer Linear Programming |
| IP | Integer Programming |
| F&O | Fix-and-Optimize |
| VNS | Variable Neighborhood Search |
| NRRP | Nurse Re-Rostering Problem |
| ALNS | Adaptive Large Neighborhood Search |
| SA | Simulated Annealing |
| NRP | Nurse Rostering Problem |
| LB | Lower Bound |
| CG | Column Generation |
| LA | Late Acceptance |
| LAFO | Late Acceptance Fix-and-Optimize |

# LIST OF FIGURES

**LIST OF TABLES**

## CONTENTS

# 1 INTRODUCTION

The construction of work schedules for nurses is an operational process found in any hospital sector that provides a nursing service. This process, hereafter referred to as the Nurse Rostering Problem (NRP), consists of assigning a given set of nurses to work shifts distributed into a planning horizon of multiple weeks. In addition to building a feasible schedule, a solution for the NRP needs to consider several requirements, such as skill-tailored assignments, labor laws, institutional norms and employee preferences. The high number of requirements associated with the combinatorial nature of the problem results in a process that could take several days to solve manually, even to produce a low-quality schedule.

The construction of a schedule is critical because it directly influences the quality of nursing care service. A deficient schedule can result in several negative consequences such as: (i) overwork, (ii) unjustified absences, (iii) illness, (iv) internal conflicts among team members, and (v) errors in nursing care that endangers the patient lives. Hence, the automation of this task by software is becoming essential to ensure the quality of the nursing services, specially in medium and large hospitals (COSTA; MORITA; MARTINEZ, 2000; ARENDT, 2010).

In addition, for being considered an NP-Hard optimization problem (OSOGAMI; IMAI, 2000), the research on the NRP is still challenging due to several problem variants proposed in the literature with a distinct set of objectives and requirements. In fact, this problem diversity makes it difficult to compare the performance of different resolution methods. In order to soften this issue, programming competitions have been organized comparing techniques and stimulating the publication of results on standardized versions of the NRP. The most recent problem version was formalized in the Second International Nurse Rostering Competition (INRC-II) (CESCHIA et al., 2019). In this competition, a *multi-stage* problem was proposed, where solutions for single weeks should be produced sequentially by the solver, without prior information about the requirements in the following weeks. While the multi-stage version is useful in a practical application, the researchers used the so called *static* version of the INRC-II problem to produce optimal solutions and evaluate the quality of the forecasting approaches. In the static version, instead of solving each week sequentially, they are solved at once.

For the static version of the INRC-II, advanced solving techniques were developed, including Integer Programming (IP) and metaheuristics (WICKERT; SARTORI; BURIOL, 2016; GOMES; TOFFOLO; SANTOS, 2017; LEGRAIN; OMER; ROSAT, 2019; CESCHIA; GUIDO; SCHAERF, 2020). It is important to highlight that, although there is a common understanding among researchers regarding the difficulty of solving the static version exactly through a MIP approach, to the best of our knowledge, there is no publication with numerical results to support this assertion. This gap motivates the investigation of the limitations of MIP approaches, including how different requirements impact the resolution of the problem and how this approach could perform when inserted in a matheuristic procedure.

In this work, we focus on the *static* version of the INRC-II problem. Our main contribution to this problem is two-fold. (i) We evaluate the performance of a state-of-the-art MIP solver when handling three different mathematical programming formulations proposed to the problem. Also, we carried out an experimental evaluation with relaxed versions of the problem that revealed the consecutive requirements are the most challenging ones to be handled by a MIP approach. Finally, we propose a novel formulation that exploits the non-uniform distribution of consecutive requirements in a dataset. (ii) We propose a matheuristic approach that combines a Late Acceptance criteria as a diversification strategy into a fix-and-optimize heuristic. Since this heuristic works by solving subproblems, we also propose a decomposition procedure that optimizes only a subset of nurses and weeks in each iteration. One of the main characteristics of the fix-and-optimize is that it can significantly improve the performance of an underlying MIP solver by achieving solutions of better quality at early stages. As we further detail in Section 4, such a procedure easily maintains and accommodates novel requirements. Finally, we present a comprehensive comparison of our results against the ones reported by state-of-the-art methods.

The remaining of this work is organized as follows: Chapter 2 provides a literature review on exact and heuristic approaches designed to solve the nurse rostering problem with the focus on the INRC-II. A formal definition of the problem tackled in this study is presented in Chapter 3 along with mathematical formulations. Chapter 4 presents a description of the Late Acceptance Fix-and-Optimize approach. In Chapter 5 we present an extensive set of experiments to evaluate and compare the performance of different approaches and models on instances of the INRC-II. Finally, Chapter presents 6 our major conclusions of this research and directions for future work.

# 2 LITERATURE REVIEW

The Nurse Rostering Problem (NRP) has several variants defined in the context of different countries. In this chapter, we first present a brief literature review comprising the main optimization methods proposed to the NRP. Next, we focus on solution approaches specifically designed to solve NRP variants proposed in the first two editions of the International Nurse Rostering Competition.

## 2.1 Solution approaches

The Nurse Rostering Problem (NRP) is part of a wide class of problems called Personnel Scheduling. Problems in this class have been studied for decades and focus on different approaches that have been resolved. A literature review on Personnel Scheduling problems published by Bergh et al. (2013) classified 291 publications according to solution methods and application areas in which the NRP appears the most common problem. For more details in the NRP, we refer the reader to the work of Burke et al. (2004).

Due to its complexity and relevance in practice, since the 1960s, papers have been published on various aspects of healthcare personnel scheduling (WOLFE; YOUNG, 1965a; WOLFE; YOUNG, 1965b). In the beginning, researchers attempted to develop mathematical models for the problem, such as the Mixed Integer Programming (MIP) formulation presented in Warner and Prawda (1972). However, they were not able to find solutions to solve real-world problems due to their complexity. In fact, most MIP approaches successfully applied for solving the NRP were designed to solve basic variants of the problem that do not consider real-world constraints as, for example, consecutive assignments. As shown by Smet (2018), adding consecutive assignments into a compact MIP model could increase the running time by $98\%$.

Hence, many heuristics methods have been proposed to solve NRP, such as Simulated Annealing (ISKEN; HANCOCK, 1991) and Tabu Search (BERRADA; FERLAND; MICHELON, 1996). Since then, several other algorithms have been used to solve the NRP, such as heuristics (OSOGAMI; IMAI, 2000), hyper-heuristics (ASTA; ÖZCAN; CURTOIS, 2016) and integer programming formulations (MASON; SMITH, 1998). More recently, Matheuristic approaches, which combine heuristics with mathematical programming, have been proposed quite frequently as a promising research direction. An example is a work of Burke, Li and Qu (2010), which aims to solve real-world instances provided by a Dutch hospital. The authors use integer programming to find a feasible initial solution that is further refined with a VNS.

Besides the large number of variants existing in the NRP, the comparison of different resolution methods has become one of the main challenges of the research area. In order to soften this problem, two editions of the *International Nurse Rostering Competition* were proposed, hereafter referred as (INRC-I) (HASPESLAGH et al., 2010) and (INRC-II) (CESCHIA et al., 2019). The main difference between them is that the INRC-I proposed a static problem, while in the INRC-II a multi-stage problem was tackled. In a *static* problem, the planning horizon is solved at once, i.e., the solution is obtained in a single stage. By contrast, in a *multi-stage* problem, the planning horizon is split into multiple stages, in which each stage must be solved separately. Hence, the final solution is a combination of the solutions produced in each stage.

Both competitions are detailed in the following sections, focusing on the second edition, whose problem is the subject of study in this work.

## 2.2 First International Nurse Rostering Competition (INRC-I)

The static problem defined in the INRC-I was sponsored by the International Conference on the Practice and Theory of Automated Timetabling (PATAT). Competitors were allowed to submit a solution approach for three tracks: sprint, medium, and long. Each track comprises a set of instances and allows the maximum computation time of 10 seconds, 10 minutes, and 10 hours. The winner was chosen based on the effectiveness of a particular technique in a specified time.

The INRC-I winner method, proposed by Valouxis et al. (2012), used a two-stage approach to decompose each instance into sub-problems, which can be solved using integer mathematical programming. Also, local search techniques were applied to test different combinations of nurse assignments.

The second-placed (BURKE; CURTOIS, 2010) developed two different algorithms. The first one is an ejection chain based method that was applied to the sprint instances. The second algorithm, a branch and price method, was applied to the medium and long instances. The computational experiments showed that the branch and price algorithm often solved these instances and provided novel results for the benchmark (BURKE; CURTOIS, 2014).

The third-placed method also includes heuristics. Bilgin et al. (2010) proposed a hyper-heuristic approach combined with a greedy shuffle heuristic. The solutions for the sprint instances were close to optimal.

After the end of the competition, Santos et al. (2014) proposed a MIP model and employed heuristic techniques to decompose the problem. Starting from a compact formulation of the problem, they applied improved cut generation strategies and primal heuristics, combining an aggressive clique separation process with a MIP heuristic. The experimental results showed they improved several best-known solutions by using this approach.

## 2.3 Second International Nurse Rostering Competition (INRC-II)

Building on the success of the INRC-I, in 2014, the same organizers announced the INRC-II. As the main difference, this second edition introduced a multi-stage problem in which each stage corresponds to a week in the planning horizon. According to the rules, it was mandatory to solve each stage separately. Consequently, the solving method should deal with the uncertainty of future data while solving consecutive weeks. The complete solution is composed and evaluated exactly only after the last week had been solved. The multi-stage problem is supposed to fit a real-world scenario in which changes in the schedule often happen, such as relocation of nurses due to absences or demand from other sectors (CAUSMAECKER et al., 2004). Besides that, the competition evaluated each team approach by using datasets with multi-skilled nurses and planning horizons of 4 and 8 weeks.

At the end of the competition, several papers were published with the aim of producing optimal solutions for INRC-II problems. In order to achieve that, instead of solving each stage of a problem instance separately, these studies have been solving all stages at once. That approach defined a static version of the INRC-II problem. Thus, in this section, we present the related work proposed to solve the INRC-II classified in multi-stage and static approaches.

### 2.3.1 Multi-stage Approaches

The first and second-placed teams employed methods based on mathematical programming. The winners, Römer and Mellouli (2016a), formulated the problem as a multicommodity flow Mixed Integer Linear Programming (MILP) based on state-expanded networks. Also, to handle the multi-stage characteristic, the authors used a deterministic lookahead approach using artificially generated demand data. The authors further studied in Römer and Mellouli (2016b) the future demand uncertainty in personnel scheduling, investigating deterministic lookahead policies using optimization and simulation.

The second-placed team was occupied by Legrain, Omer and Rosat (2018). The authors proposed an algorithm that embeds a primal-dual algorithm within a sample average approximation. The primal-dual procedure generates candidate schedules for the current week, and the average sample approximation allows evaluating each of them retaining the best one. The stochastic algorithm performed very well over a wide range of instances of the competition, although it was not able to find the best objective value or optimal solutions in most cases.

Kheiri et al. (2016) proposed a selection hyper-heuristic framework that mixes and controls a set of nine low-level heuristics to solve the INRC-II problem. The method was ranked third, and it is reliable in terms of producing feasible solutions. The Ortec company (Netherlands), (JIN; POST; VEEN, 2016), added some artificial soft constraints in the optimization commercial solver to steer a good connection between two consecutive weeks and achieve high-quality solutions.

Mischek and Musliu (2019) proposed and evaluated several original extensions of basic IP formulations for the INRC-II problem in order to deal with multi-stage settings. They showed that the extensions improved upon the results of the basic model and achieved competitive results compared to the finalists in the competition.

### 2.3.2 Static Approaches

In Wickert, Sartori and Buriol (2016), the authors focused on the static version of the INRC-II problem. They presented a MILP model and a fix-and-optimize matheuristic combined with a VNS method. The developed approach uses four different decompositions: week, nurse, day, and shift. The method generates feasible solutions for all instances within the time limits of the competition. Also, the average results were better than the ranked second, which runs a multi-stage version of the problem. The authors further studied different strategies for the Nurse Re-Rostering Problem (WICKERT; SMET; BERGHE, 2019). The research is based on relaxing some soft constraints and the rescheduling periods. They proposed a general IP formulation and a Variable Neighborhood Descent heuristic. The solution methods were tested on the instances adapted from the INRC-II.

Legrain, Omer and Rosat (2019) modeled the INRC-II problem as an integer program solved using a branch-and-price algorithm based on rotations. To achieve good results on large instances, they developed an ALNS behaving as a primal heuristic in interaction with the branch-and-price. Initial solutions of the ALSN were found by a rolling-horizon algorithm well-suited to the rotation model. The approach was tested on some instances of the INRC-II, and when compared with the best solutions found in the multi-stage version, they achieved an overall 15.2% improvement.

The static version also was studied by Gomes, Toffolo and Santos (2017). They presented a VNS accelerated column generation procedure for the INRC-II problem in addition to a relax-and-fix heuristic for obtaining feasible solutions. The authors showed that combining the method with heuristics significantly accelerates the method's convergence, thereby improving 29 hidden INRC-II instances with 4-week scheduling horizons.

More recently, Ceschia, Guido and Schaerf (2020) proposed a Simulated Annealing (SA) method based on a composition of large neighborhoods for the static version of the INRC-II problem. The search method uses a combination of neighborhoods that either move swaps the shifts of two nurses, or changes the shift and/or the skill assigned to a nurse for multiple consecutive days. Computational results showed that they obtained results that improve on all previous approaches on some data sets for the longer time limit. When a shorter time limit was used, the method provided better results than the literature on most instances.

# 3 NURSE ROSTERING PROBLEM

In this chapter, we introduce the Nurse Rostering Problem (NRP) that was previously defined in the second version of the International Nurse Rostering Competition (INRC-II). Section 3.1 presents an overview of the NRP, as well as the requirements considered in the problem. In Section 3.2, the NRP is formally defined using three mathematical models proposed in this work.

## 3.1 Problem definition

The goal of the problem is to build a schedule for a set of nurses $N$ considering a planning horizon composed of a set of days $D$, in which each day is split into a set of shifts $S$. Each nurse $n \in N$ has one or more skills from a set $K$. In order to fulfill a schedule, one needs to assign, for each nurse, a shift and a skill, for all days in the planning horizon, by taking into account a given set of requirements.

For each nurse schedule, we introduce a set of definitions related to its assignments in the planning horizon. We call as a *work shift* the shift in which a nurse is assigned in a given day $d \in D$. A given day $d$ in the schedule of a nurse is a *work day* if it has an associated work shift, otherwise the day $d$ is called a *day off*. A *work weekend* occurs when a nurse has at least one work day in a weekend. In other words, it occurs when it has a work day on Saturday or Sunday, or both. Particularly, when a work weekend has two work days, we call it a *complete weekend*.

As a matter of example, Figure 3.1 displays a toy instance of the NRP composed by a short planning horizon with only three days (Friday, Saturday, Sunday), six nurses ($N_1$, $N_2$, $N_3$, $N_4$, $N_5$, $N_6$), four shifts (Early, Day, Late, Night), and three skills (Head Nurse (H), Nurse (N), Trainee (T)). Each cell in this table represents a working assignment. The symbol "–" indicates a day off. In this instance, we can observe that, for example, the nurse $N_1$ is assigned to the Late shift on Friday as a HeadNurse and the Night shift on Saturday as a Nurse. On Sunday, this same nurse has a day off. All nurses have two work days, except $N_4$ that has a single one with a Night work shift and $N_5$ that has three work days. Finally, we also note that while the nurse $N_4$ is the only one without a work weekend, the nurses $N_2$, $N_5$, and $N_6$ are the only ones with a complete weekend.

Figure 3.1: Example of a toy instance with three days, six nurses, four shifts and three skills.

| Nurse | Friday | Saturday | Sunday |
|---|---|---|---|
| $N_1$ | Late [H] | Night [N] | – |
| $N_2$ | – | Early [T] | Early [T] |
| $N_3$ | Day [N] | – | Day [H] |
| $N_4$ | Night [N] | – | – |
| $N_5$ | Early [T] | Day [T] | Night [T] |
| $N_6$ | – | Late [H] | Late [H] |

Source: created by author.

The main requirements of a schedule are determined by two parameters: *minimum coverage* and *optimal coverage*. While the minimum coverage defines the number of nurses of each skill required for the minimum operation of the health care service, the optimal coverage defines the number of nurses required to operate the service under optimal conditions. Usually, both parameters are statistically set for each day and shift by the hospital staff based on historical demands.

In fact, for solving the NRP, one needs to consider several more requirements, split into two categories: *hard requirements* and *soft requirements*. While hard requirements must be satisfied in order to produce a *feasible* solution, soft requirements should be satisfied as much as possible since they directly influence the solution quality. For each violation of soft requirements, a weighted cost is penalized in the objective function. Both types of requirements of the INRC-II problem are shown as following and a detailed description can be found in Ceschia et al. (2019).

Hard requirements:

$H1$ **Single assignment per day:** each nurse can only work in a single shift per day by using a single skill.

$H2$ **Insufficient staffing for minimum coverage:** the number of nurses of a given skill, assigned to a given shift on a day, must respect the minimum coverage.

$H3$ **Shift type successions:** a shift type succession must belong to a valid succession (e.g., a night shift cannot be followed by a morning shift).

$H4$ **Missing required skill:** nurses cannot be assigned to work with a skill they do not have.

Soft requirements:

$S1$ **Insufficient staffing for optimal coverage:** the number of nurses of a given skill, assigned to a given shift on a day should attend the optimal coverage. For each nurse missing to reach the optimal coverage value, there is a cost. The amount of nurses exceeding the optimal coverage are not penalized.

$S2$ **Consecutive assignments:** the following limits should be attended for each nurse:

a) Minimum number of consecutive work days.
b) Maximum number of consecutive work days.
c) Minimum number of consecutive assignments by shift.
d) Maximum number of consecutive assignments by shift.

$S3$ **Consecutive days off:** the following limits should be attended for each nurse:

a) Minimum number of consecutive days off.
b) Maximum number of consecutive days off.

$S4$ **Assignment preferences:** a nurse should not be assigned to:

a) An undesired work shift.
b) An undesired work day.

$S5$ **Complete weekend:** a nurse should work on both days of a weekend, or none.

$S6$ **Total assignments:** the following limits should be satisfied for each nurse considering the whole planning horizon:

a) Maximum number of work days.
b) Minimum number of work days.

$S7$ **Total work weekends:** to each nurse is associated a maximum number of work weekends that should be respected.

Note that the requirements $H3$, $S2$, and $S3$ need the information of a previous schedule to be evaluated in the static problem. This data is provided in the *history file*.

## 3.2 Integer Programming Formulation

In this section, we present three integer programming formulations for the NRP, hereafter referred as models $M_1$, $M_2$, $M_3$, by extending a base model denoted as $M_0$. The mathematical notation used in all models is presented in detail in Tables 3.1, 3.2 and 3.3.

Table 3.1: Sets used in the mathematical formulations.

| **Sets** | Definition |
|---|---|
| $n \in N$ | set of nurses. $N = \{1, ..., \|N\|\}$ |
| $d \in D$ | set of days. $D = \{1, ..., \|D\|\}$ |
| $d \in D^{-i}$ | set of days $d \in D$ such as $d \leq \|D\| - i$ |
| $(i, j) \in W$ | set of weekends such that $i, j \in D$. |
| $s \in S$ | set of shifts. $S = \{1, ..., \|S\|\}$. We refer to a day off as a last shift $s = \|S\|$. |
| $s \in S'$ | set of shifts without day off. $S' = \{1, ..., \|S\| - 1\}$ |
| $k \in K$ | set of skills. $K = \{1, ..., \|K\|\}$ |
| $(s_1, s_2) \in F$ | set of pairs with invalid shift successions. A pair $(s_1, s_2)$ means that the shift $s_2 \in S'$ cannot be followed by the shift $s_1 \in S'$. |
| $(u, v) \in B$ | set of block assignments. Each pair $(u, v)$ represents a possible consecutive assignment starting on day $u \in D$ and ending on day $v \in D$, such that $u \leq v$. |

Table 3.2: Parameters used in the mathematical formulations.

| **Parameters** | Definition |
|---|---|
| $T_{nk} \in \{0,1\}$ | 1 if nurse $n$ has skill $k$, 0 otherwise. |
| $P_{nds} \in \{0,1\}$ | 1 if nurse $n$ does not prefer to work in the shift $s$ on day $d$, 0 otherwise. |
| $P_{nd} \in \{0,1\}$ | 1 if nurse $n$ prefers a day off on day $d$, 0 otherwise. |
| $V^{-}_{dsk} \in \mathbb{N}^*$ | minimum coverage value for day $d$, shift $s$ and skill $k$. |
| $V^{*}_{dsk} \in \mathbb{N}^*$ | optimal coverage value for day $d$, shift $s$ and skill $k$. |
| $L^{+}_{n} \in \mathbb{N}^*$ | maximum number of consecutive work days for nurse $n$. |
| $L^{-}_{n} \in \mathbb{N}^*$ | minimum number of consecutive work days for nurse $n$. |
| $L^{+}_{ns} \in \mathbb{N}^*$ | maximum number of consecutive assignments for shift $s \in S$ and nurse $n$. |
| $L^{-}_{ns} \in \mathbb{N}^*$ | minimum number of consecutive assignments for shift $s \in S$ and nurse $n$. |
| $V_{n} \in \{0,1\}$ | 1 if the nurse $n$ prefers to complete weekends. |
| $Q^{+}_{n} \in \mathbb{N}^*$ | maximum number of work days for nurse $n$. |
| $Q^{-}_{n} \in \mathbb{N}^*$ | minimum number of work days for nurse $n$. |
| $R^{+}_{n} \in \mathbb{N}^*$ | maximum number of work weekends for nurse $n$. |
| $W^{S1}..W^{S7} \in \mathbb{N}^*$ | requirement weights specified by INRC-II: $W^{S1}$=30; $W^{S2_{ab}}$=30; $W^{S4}$=10; $W^{S5}$=30; $W^{S6}$=20; $W^{S7}$=30; |
| $W^{S2S3}_{s} \in \mathbb{N}^*$ | weights of the requirements $S2_{cd}$ and $S3_{ab}$ according to shift $s \in S$ specified by INRC-II: $W^{S2S3}_{s}$=15 if $s \in S'$, and $W^{S2S3}_{s}$=30 if $s$=$\|S\|$. |
| **Constant** | |
| $M$ | constant with value 10. |

Table 3.3: Variables used in the mathematical formulations.

| **Variables** | Definition |
|---|---|
| **Decision Variable** | |
| $x_{ndsk} \in \{0,1\}$ | 1 if nurse $n$ is assigned to shift $s \in S$ with skill $k$ on day $d$, 0 otherwise. |
| **Aux. Variables** | |
| $r_{nij} \in \{0,1\}$ | 1 if a nurse $n$ has a work weekend $(i,j) \in W$. |
| $b_{nuv} \in \{0,1\}$ | 1 if a nurse $n$ starts a block of consecutive assignments beginning on day $u$ and ending on day $v$. |

| | |
|---|---|
| $b_{nuvs} \in \{0,1\}$ | 1 if a nurse $n$ starts a block of consecutive assignments with shift $s$ beginning on day $u$ and ending on day $v$. |
| $g_{nd} \in \{0,1\}$ | 1 if the number of consecutive work days exceeds $L^+_n$ for nurse $n$ on day $d$. |
| $g_{nds} \in \{0,1\}$ | 1 if the number of consecutive work shifts $s$ exceeds $L^+_{ns}$ for nurse $n$ on day $d$. |
| $m^D_{ndt} \in \{0,1\}$ | $t \in \{1,2,3,4\}$, 1 if the number of consecutive work days $(d..d+t)$ is a sequence that penalize the minimum $L^-_n$. Ex: Minimum of 3 consecutive work days ($L^-_n = 3$), $md_{nd3} = 1$ for the sequence $(110)$. |
| $m^S_{ndst} \in \{0,1\}$ | $t \in \{1,2,3,4\}$, 1 if the number of consecutive shifts days $(d..d+t)$ is a sequence that penalize the minimum $L^-_{ns}$. Similar $md$. |
| $t_{nd} \in \{0,1\}$ | 1 if nurse $n$ starts a block with consecutive assignments beginning on day $d$. |
| $t_{nds} \in \{0,1\}$ | 1 if nurse $n$ starts a block with consecutive assignments with shift $s$ beginning on day $d$. |
| $C^{S1}_{dsk} \in \mathbb{N}^*$ | missing nurses to reach the optimal coverage $V^*_{dsk}$. |
| $a_{ndt} \in \mathbb{N}_0$ | number of consecutive work days for nurse $n$ starting on day $d$ and pattern $t$. Ex: Minimum of 3 consecutive work days. When $t = 1$ the pattern is (010), and when $t = 2$ the pattern is (0110). If there is a pattern, $a_{ndt} = t$. |
| $a_{ndst} \in \mathbb{N}_0$ | number of consecutive shift days $s$ for nurse $n$ starting on day $d$ and pattern $t$. Similar to $a_{ndt}$. |
| $c_{ndt} \in \mathbb{N}_0$ | number of missing days to reach the minimum consecutive work days for nurse $n$ starting on day $d$ and pattern $t$. |
| $c_{ndst} \in \mathbb{N}_0$ | number of missing shifts $s$ to reach the minimum consecutive shifts day for nurse $n$ starting on day $d$ and pattern $t$. |
| $C^{S2_a}_{nij} \in \mathbb{N}^*$ | missing days to reach $L^-_n$ in the block $(i,j) \in B$ for nurse $n$. |
| $C^{S2_c S3_a}_{nijs} \in \mathbb{N}^*$ | missing shifts to reach $L^-_{ns}$ in the block $(i,j) \in B$ for nurse $n$ and shift $s$. |
| $C^{S5}_{nij} \in \{0,1\}$ | 1 if a nurse $n$ has exactly a single work day in the weekend $(i,j) \in W$ violating complete weekend requirement $V_n$. |
| $C^{S6_a}_n \in \mathbb{N}^*$ | number of work days of nurse $n$ that exceeds $Q^+_n$. |
| $C^{S6_b}_n \in \mathbb{N}^*$ | missing days for nurse $n$ to reach $Q^-_n$. |
| $C^{S7}_n \in \mathbb{N}^*$ | number of work weekends of nurse $n$ that exceeds $R^+_n$. |

### 3.2.1 Base Model $M_0$

In this section, we present an IP formulation for the NRP considering all the hard and soft requirements mentioned in Section 3.1 except those concerning consecutive minimums (S2a, S2c, and S3a). This model, denoted as $M_0$, is similar to proposed by Wickert, Sartori and Buriol (2016). $M_0$ is further extended in the models $M_1$, $M_2$ and $M_3$.

**Minimize** (3.1)

$$Z_0 = W^{S1} \sum_{d\in D} \sum_{s\in S'} \sum_{k\in K} C^{S1}_{dsk} + W^{S2_{ab}} \sum_{n\in N} \sum_{d\in D} g_{nd} + \sum_{s\in S} W^{S2S3}_{s} \sum_{n\in N} \sum_{d\in D} g_{nds} + W^{S4} \sum_{n\in N} \sum_{d\in D} \sum_{s\in S'} \sum_{k\in K} (x_{ndsk}P_{nds} + x_{ndsk}P_{nd}) + W^{S5} \sum_{n\in N} \sum_{(i,j)\in W} C^{S5}_{nij} + W^{S6} \sum_{n\in N} (C^{S6_a}_{n} + C^{S6_b}_{n}) + W^{S7} \sum_{n\in N} C^{S7}_{n}$$

**Subject to**

$$\sum_{s\in S} \sum_{k\in K} x_{ndsk} = 1 \qquad \forall n \in N, d \in D \qquad (3.2)$$

$$\sum_{n\in N} x_{ndsk} \geq V^{-}_{dsk} \qquad \forall d \in D, s \in S', k \in K \qquad (3.3)$$

$$\sum_{k\in K} (x_{n,d-1,s_1,k} + x_{n,d,s_2,k}) \leq 1 \qquad \forall n \in N, d \in D \setminus \{1\}, (s_1, s_2) \in F \qquad (3.4)$$

$$\sum_{s\in S'} x_{ndsk} \leq T_{nk} \qquad \forall n \in N, d \in D, k \in K \qquad (3.5)$$

$$C^{S1}_{dsk} \geq V^{*}_{dsk} - \sum_{n\in N} x_{ndsk} \qquad \forall d \in D, s \in S', k \in K \qquad (3.6)$$

$$\sum_{k\in K} \sum_{i\in d..d+L^{+}_{ns}} x_{nisk} \leq L^{+}_{ns} + g_{nds} \qquad \forall n \in N, d \in D, s \in S{:}d{\leq}|D|{-}L^{+}_{ns} \qquad (3.7)$$

$$\sum_{s\in S'} \sum_{k\in K} \sum_{i\in d..d+L^{+}_{n}} x_{nisk} \leq L^{+}_{n} + g_{nd} \qquad \forall n \in N, d \in D : d \leq |D| - L^{+}_{n} \qquad (3.8)$$

$$2r_{nij} \geq \sum_{s\in S'} \sum_{k\in K} (x_{nisk} + x_{njsk}) \qquad \forall n \in N, (i,j) \in W \qquad (3.9)$$

$$C^{S7}_{n} \geq \sum_{(i,j)\in W} r_{nij} - R^{+}_{n} \qquad \forall n \in N \qquad (3.10)$$

$$C^{S5}_{nij} = (2r_{nij} - \sum_{s\in S'} \sum_{k\in K} (x_{nisk} + x_{njsk}))V_{n} \qquad \forall n \in N, (i,j) \in W \qquad (3.11)$$

$$C^{S6_a}_{n} \geq \sum_{d\in D} \sum_{s\in S'} \sum_{k\in K} x_{ndsk} - Q^{+}_{n} \qquad \forall n \in N \qquad (3.12)$$

$$C_n^{S6_b} \geq Q_n^- - \sum_{d\in D}\sum_{s\in S'}\sum_{k\in K} x_{ndsk} \qquad \forall n \in N \tag{3.13}$$

$$x_{ndsk}, g_{nd}, g_{nds}, r_{nij}, C_{nij}^{S5} \in \{0,1\} \qquad \forall n \in N, d \in D, s \in S, k \in K, (i,j) \in W$$

$$C_n^{S6_a}, C_n^{S6_b}, C_n^{S7}, C_{dsk}^{S1} \in \mathbb{N}^* \qquad \forall n \in N, s \in S, k \in K$$

The objective function of the model is composed of several weighted parts presented by Equation (3.1). While requirement $S4$ is computed directly by variable in the objective function, the remaining soft requirements also require the inclusion of constraints sets in the formulation.

Constraint set (3.2) ensures that a nurse is assigned to one work shift or a day off for each day. Constraint set (3.3) ensures that minimum coverage is respected. Constraint set (3.4) guarantee that shift successions are valid, according to requirement $H3$. Constraint set (3.5) ensures that a nurse has a work shift assigned among its skillset. Constraint set (3.6) computes the cost variable $C_{dsk}^{S1}$, the number of missing nurses for optimal coverage, regarding the requirement $S1$. Constraints sets (3.7)-(3.8) store on $g$ variables the number of consecutive assignments, concerning work shifts, work days, and days off, that exceeds the limit established by the corresponding parameter $L^+$.

Constraint set (3.9) enforces the variable $r_{nij}$ to be equal to one when a nurse works at least one day of the weekend $(i,j) \in W$. This variable is used in the constraint set (3.10) to determine the cost variable $C_n^{S7}$, i.e., how many work weekends are exceeding the limit $R_n^+$ for each nurse. The variable $r_{nij}$ is also used in the constraint set (3.11) that sets the variables $C_{nij}^{S5}$ related to the requirement $S5$. It ensures that the variable $C_{nij}^{S5}$ will be activated only when the nurse $n$ has exactly a single work day in the weekend $(i,j) \in W$ and it violates complete weekend requirement $V_n$.

Constraints sets (3.12) and (3.13) concern the maximum and minimum limits of work days for each nurse regarding the whole schedule. While constraint set (3.12) computes the cost variable $C_n^{S6a}$ as the number of workdays that exceeds $Q_n^+$, constraint set (3.13) computes the cost variable $C_n^{S6b}$ as the number of workdays below $Q_n^-$.

For the sake of simplicity, all constraints related to a previous schedule ($H3$, $S2$ and $S3$) are not explicitly described here. Note that this could be handled by prepending a week where all allocations are fixed according to the history file.

### 3.2.2 Model $M_1$

In this section, we present an IP formulation denoted as $M_1$ that extends the base model $M_0$ by adding the soft requirements S2a, S2c, and S3a. These additional requirements are formulated by using the same approach presented in the work of Dorneles, Araújo and Buriol (2012) in which consecutive assignments are identified as blocks. The model $M_1$ is described below.

**Minimize** (3.14)

$$Z = Z_0 + W^{S2_{ab}} \sum_{n \in N} \sum_{(i,j) \in B} C_{nij}^{S2_a} + \sum_{s \in S} W_s^{S2S3} \sum_{n \in N} \sum_{(i,j) \in B} C_{nijs}^{S2_c S3_a}$$

**Subject to**

Constraints sets (3.2)-(3.10)

$$b_{nuv} \leq \sum_{s \in S', k \in K} x_{nisk} \qquad \forall n \in N, (u,v) \in B, i \in u..v \tag{3.15}$$

$$b_{nuv} + \sum_{s \in S', k \in K} x_{n,u-1,s,k} \leq 1 \qquad \forall n \in N, (u,v) \in B : u > 1 \tag{3.16}$$

$$b_{nuv} + \sum_{s \in S', k \in K} x_{n,v+1,s,k} \leq 1 \qquad \forall n \in N, (u,v) \in B : v < |D| \tag{3.17}$$

$$\sum_{d \in D, s \in S', k \in K} x_{ndsk} = \sum_{(u,v) \in B} (v-u+1) b_{nuv} \qquad \forall n \in N \tag{3.18}$$

$$b_{nuvs} \leq \sum_{k \in K} x_{nisk} \qquad \forall n \in N, s \in S, (u,v) \in B, i \in u..v \tag{3.19}$$

$$b_{nuvs} + \sum_{k \in K} x_{n,u-1,s,k} \leq 1 \qquad \forall n \in N, s \in S, (u,v) \in B : u > 1 \tag{3.20}$$

$$b_{nuvs} + \sum_{k \in K} x_{n,v+1,s,k} \leq 1 \qquad \forall n \in N, s \in S, (u,v) \in B : v < |D| \tag{3.21}$$

$$\sum_{d \in D, k \in K} x_{ndsk} = \sum_{(u,v) \in B} (v-u+1) b_{nuvs} \qquad \forall n \in N, s \in S \tag{3.22}$$

$$C_{nuv}^{S2_a} \geq L_n^- b_{nuv} - (v-u+1) b_{nuv} \qquad \forall n \in N, (u,v) \in B : u > 1 \tag{3.23}$$

$$C_{nuvs}^{S2_c S3_a} \geq L_{ns}^- b_{nuvs} - (v-u+1) b_{nuvs} \qquad \forall n \in N, (u,v) \in B, s \in S : v < |D|, u > 1 \tag{3.24}$$

$$x_{ndsk}, b_{nij}, b_{nijs} \in \{0,1\} \qquad \forall n \in N, (i,j) \in B, s \in S, k \in K$$

$$C_{nij}^{S2_a}, C_{nijs}^{S2_c, S3_a} \in \mathbb{N}^* \qquad \forall n \in N, s \in S, (i,j) \in B$$

Constraints sets (3.15)-(3.18) ensure that the variable $b_{nuv}$ will only be activated when the nurse $n$ has a block $(u,v)$ of consecutive work days. This auxiliary variable is used to compute the soft requirements $S2_a$, $S2_c$ and $S3_a$. Similarly, the constraints sets (3.19)-(3.22) ensure that the variable $b_{nuvs}$ will only be activated when the nurse $n$ has a block $(u,v)$ composed by consecutive assignments of the shift $s$.

Constraint set (3.23) determines the number of missing days to reach the minimum number of consecutive work days (requirement $S2_a$). Similarly, constraint set (3.24) determines the number of missing shifts to reach the minimum number of consecutive assignments by shift (requirements $S2_c$ and $S3_a$).

### 3.2.3 Model $M_2$

The model $M_2$ extends the model $M_0$ 3.2.1 by adding the soft requirements S2a, S2c and S3a with the same formulation proposed by Wickert, Sartori and Buriol (2016) and further detailed in Wickert (2019). This model identifies patterns according to the number of minimum consecutive requirements. For example, considering four as the minimum number of consecutive working days, 1 as a work day, and 0 as a day off, the search patterns are 010, 0110, 01110. If found, these patterns represent three, two and one violations, since they are less than four consecutive working days. The model $M_2$ is described below.

**Minimize** (3.25)

$$Z = Z_0 + W^{S2_{ab}} \sum_{n \in N} \sum_{d \in D} \sum_{t \in 1..L_d^- - 1} c_{ndt} + \sum_{s \in S} W_s^{S2S3} \sum_{n \in N} \sum_{d \in D} \sum_{t \in 1..L_d^- - 1} c_{ndst}$$

**Subject to**

Constraints sets (3.2)-(3.10)

$$a_{ndt} + c_{ndt} \geq L_n^- \qquad \forall n \in N, t \in 1..L_n^- - 1, d \in 1..|D| - (t+1) \qquad (3.26)$$

$$a_{ndt} = \sum_{i \in d..(t+d+1), s \in S, k \in K} x_{nisk} + \sum_{s \in S', k \in K} (x_{ndsk} + x_{n,t+d+1,sk})M + \sum_{j \in (d+1)..(t+d)} (1 - \sum_{s \in S', k \in K} x_{njsk})M \qquad \forall n \in N, t \in 1..L_n^- - 1, d \in 1..|D| - (t+1) \qquad (3.27)$$

$$a_{ndst} + c_{ndst} \geq L^{-}_{ns} \qquad \forall n \in N, s \in S, t \in 1..L^{-}_{ns} - 1, d \in 1..|D| - (t+1) \quad (3.28)$$

$$a_{ndst} = \sum_{i \in d..(t+d+1), k \in K} x_{nisk} + \sum_{k \in K} (x_{ndsk} + x_{n,t+d+1,sk})M + \sum_{j \in (d+1)..(t+d)} (1 - \sum_{k \in K} x_{njsk})M \qquad \forall n \in N, s \in S, t \in 1..L^{-}_{ns} - 1, d \in 1..|D| - (t+1) \quad (3.29)$$

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

$$a_{ndt}, c_{ndt} \in \mathbb{N}_0 \qquad \forall n \in N, d \in D, t \in 1..L^{-}_{n} - 1$$

$$a_{ndts}, c_{ndts} \in \mathbb{N}_0 \qquad \forall n \in N, d \in D, s \in S, t \in 1..L^{-}_{ns} - 1$$

Constraint sets (3.26) and (3.27) compute the minimum consecutive assignments of working days violations and store the result in the variable $c_{ndt}$. The right side of the constraint set (3.27) has three components that are used to match a pattern. The first component calculates the sum of the working days, the second one verifies two border bits multiplied by M, and finally, the last component is the complement of middle bits multiplied by M. For example, if the minimum are 4 consecutive working days ($L^{-}_{n} = 4$) and in the scheduling we have found the pattern 0110, it means that there are 2 violations. The number of violations is calculated as $2 + (0 * M) + (0 * M) + c_{nd2} \geq 4$ and, as result $c_{nd2}$ will assume value 2. Similarly, constraints sets (3.28) and (3.29) calculate the minimum number of consecutive assignments that occurs in the same shift, as well as days off violations. Recall that the $M$ in this formulation is a big-M.

### 3.2.4 Model $M_3$

Although the models $M_1$ and $M_2$ described in the previous section are able to model the NRP properly, empirical experiments demonstrated they have low performance when solving instances of NRP. In order to investigate this issue, we carried out a deeper analysis of individual requirements. In a separate experiment, we find out that by disregarding the requirements $S2_a, S2_c$ and $S3_a$, i.e., requirements concerning a minimum number of consecutive assignments, most instances could be solved in a few seconds.

In order to improve on this issue, we analyzed the occurrences of consecutive minimum requirements in the dataset of hidden instances of the INRC-II that are presented in detail in Section 5.1. These requirements, namely, the minimum consecutive work days ($S2_a$), days off ($S3_a$), and work shifts ($S2_c$) are defined for each nurse, and their amounts vary according to three different scenarios, as shown in Table 3.4. In this table the column *requirement* presents the requirements with the identifier names $S2_a$, $S3_a$, and $S2_c$. Column *Min* shows the possible minimum values, 2, 3, 4, and 5, that were observed in the dataset for each requirement. Finally, columns *Scenario 1*, *Scenario 2* and *Scenario 3* corresponding to scenarios with 35, 70, and 110 nurses, respectively, and present the percentage of nurses for each requirement and each minimum value that occur in the data.

Table 3.4: Percentage of nurses that are constrained with each value of the minimum requirement parameter.

| Requirement | Min | Scenario 1 (35 nurses) | Scenario 2 (70 nurses) | Scenario 3 (110 nurses) |
|---|---|---|---|---|
| Minimum consecutive work shifts ($S2_c$) | 2 | 75% | 75% | 75% |
| | 3 | 0% | 0% | 0% |
| | 4,5 | 25% | 25% | 25% |
| Minimum consecutive work days ($S2_a$) | 2 | 0% | 0% | 22% |
| | 3 | 100% | 72% | 50% |
| | 4,5 | 0% | 28% | 28% |
| Minimum consecutive days off ($S3_a$) | 2 | 60% | 72% | 44% |
| | 3 | 40% | 0% | 56% |
| | 4,5 | 0% | 28% | 0% |

Source: created by author.

According to Table 3.4, we observed that the sum of occurrences of values of minimum parameters 2 and 3 are greater than 72% in all scenarios. While the percentages of minimum $\geq 4$ have a maximum usage of 28%. This data suggests that we could explore a specialized formulation for handling each minimum value separately with a focus on providing a lightweight formulation for the parameters with values 2 and 3. This model, denoted as $M_3$, extends the base model $M_0$, and it is presented below.

**Minimize** (3.30)

$$Z = Z_0 + W^{S2_{ab}} \sum_{t \in 1..4} \sum_{n \in N : L_n^- > t} \sum_{d \in D:} m_{ndt}^D * (L_n^- - t) +$$

$$\sum_{s \in S} W_s^{S2S3} \sum_{t \in 1..4} \sum_{n \in N : L_{ns}^- > t} \sum_{d \in D:} m_{ndt}^S * (L_{ns}^- - t)$$

**Subject to**

Constraints sets (3.2)-(3.10)

$$\sum_{s \in S'} \sum_{k \in K} (x_{ndsk} - x_{n,d-1,s,k}) \leq t_{nd} \qquad \forall n \in N, d \in D : d > 1 \qquad (3.31)$$

$$\sum_{k\in K}(x_{ndsk} - x_{n,d-1,s,k}) \le t_{nds} \qquad \forall n \in N, s \in S, d \in D : d > 1 \tag{3.32}$$

$$t_{nd} - \sum_{s\in S',k\in K} x_{n,d+1,s,k} \le m^{D}_{nd1} \qquad \forall n \in N, d \in D^{-1}\text{: } L^{-}_{n} \ge 2 \tag{3.33}$$

$$t_{nd} + \sum_{s\in S',k\in K} (x_{n,d+1,s,k} - x_{n,d+2,s,k}) - 1 \le m^{D}_{nd2} \qquad \forall n \in N, d \in D^{-2}\text{: } L^{-}_{n} \ge 3 \tag{3.34}$$

$$t_{nd} - t_{n,d+2} + \sum_{s\in S',k\in K} (x_{n,d+2,s,k} - x_{n,d+3,s,k}) - 1 \le m^{D}_{nd3} \qquad \forall n \in N, d \in D^{-3}\text{: } L^{-}_{n} \ge 4 \tag{3.35}$$

$$t_{nd} - t_{n,d+2} - t_{n,d+3} + t_{n,d+4} + \sum_{s\in S',k\in K} (x_{n,d+3,sk} - x_{n,d+4,sk}) - 1 \le m^{D}_{nd4} \qquad \forall n \in N, d \in D^{-4}\text{: } L^{-}_{n} \ge 5 \tag{3.36}$$

$$t_{nds} - \sum_{k\in K} x_{n,d+1,s,k} \le m^{S}_{nds1} \qquad \forall n \in N, s \in S, d \in D^{-1}\text{: } L^{-}_{ns} \ge 2 \tag{3.37}$$

$$t_{nds} + \sum_{k\in K} (x_{n,d+1,s,k} - x_{n,d+2,s,k}) - 1 \le m^{S}_{nds2} \qquad \forall n \in N, s \in S, d \in D^{-2}\text{: } L^{-}_{ns} \ge 3 \tag{3.38}$$

$$t_{nds} - t_{n,d+2,s} + \sum_{k\in K} (x_{n,d+2,s,k} - x_{n,d+3,s,k}) - 1 \le m^{S}_{nds3} \qquad \forall n \in N, s \in S, d \in D^{-3}\text{: } L^{-}_{ns} \ge 4 \tag{3.39}$$

$$t_{nds} - t_{n,d+2,s} - t_{n,d+3,s} + t_{n,d+4,s} + \sum_{k\in K} (x_{n,d+3,sk} - x_{n,d+4,sk}) - 1 \le m^{S}_{nds4} \qquad \forall n \in N, s \in S, d \in D^{-4}\text{: } L^{-}_{ns} \ge 5 \tag{3.40}$$

$$m^{D}_{ndt}, m^{S}_{ndst}, t_{nd}, t_{nds} \in \{0,1\} \qquad \forall t \in 1..4, n \in N, d \in D, s \in S \tag{3.41}$$

The objective function $Z$ (3.30) is composed of $Z_0$ from model $M_0$ and the cost of penalization of minimum consecutive requirements.

Constraint set (3.31) ensures that the variable $t_{nd}$ will only be activated when the nurse $n$ has a block of consecutive work days assignments starting on day $d$. Similarly, constraint set (3.32) ensures that the variable $t_{nds}$ will only be activated when the nurse $n$ has a block of consecutive work shifts assignments $s$ starting on day $d$.

Constraint set (3.33) ensures that the variable $m^{D}_{nd1}$ is one when the pattern (10) is found, that is, the nurse $n$ is working on the day $d$ and the next day is a day off. In other words, when the minimum consecutive days ($L^{-}_{n}$) is 2, 3, 4 or 5 and the pattern (10) is found, $m^{D}_{nd1}$ is penalized in the objective function. Similarly, constraints sets (3.34), (3.35) and (3.36) ensure that the variables $m^{D}_{nd2}$, $m^{D}_{nd3}$ and $m^{D}_{nd4}$ are one when the patterns (110), (1110) and (11110) are found, respectively. Similarly, constraints sets (3.37)-(3.40) ensure that the variables $m^{D}_{nds}$ are one when the analogous patterns are found to shifts.

# 4 A LATE ACCEPTANCE FIX-AND-OPTIMIZE APPROACH

In this chapter, we propose a matheuristic approach hereafter defined as LAFO. This approach combines Late Acceptance criteria into a fix-and-optimize heuristic for solving instances of the INRC-II. Section 4.1 and 4.2 present, respectively, an overview of the LAFO heuristic and a detailed description of the proposed algorithm.

## 4.1 Approach overview

The base method of our approach is the fix-and-optimize heuristic that was proposed independently by Gintner, Kliewer and Suhl (2005) and by Pochet and Wolsey (2006). In the latter, the method was called *exchange*, designed to improve the relax-and-fix heuristic (WOLSEY, 1998). The main goal of the fix-and-optimize is to solve a series of subproblems that can be optimized relatively fast by a MIP solver when compared with a full problem. In each iteration of the algorithm, a decomposition process is applied to fix most of the decision variables at their value in a current solution generating a lightweight subproblem. At next, a standard MIP solver is used to solve the subproblem. The obtained solution of the subproblem becomes the current solution in case it improves the objective value. In further iterations of the algorithm new subproblems are created by fixing a different subset of variables. This process is repeated until a termination condition is satisfied. In order to improve the performance of the algorithm, usually, a time limit is imposed for the resolution of each subproblem.

In the standard F&O approach, in each iteration, the current solution is accepted only when it improves the objective value. Instead, in this work, we use the Late Acceptance criteria (LA) mechanism in order to introduce a diversification strategy into F&O. The LA criteria compares a candidate solution with a solution that was generated a given number of iterations before. If the cost of the candidate solution is better than this old solution, it will be accepted as the current one. This approach helps the algorithm escape from local optimum values by accepting temporarily worse solutions than the best found so far. This acceptance strategy is also used in the Late Acceptance Hill Climbing metaheuristic (BURKE; BYKOV, 2008).

Another critical decision in a F&O heuristic is the way decompositions are implemented. In this work, we propose a hybrid decomposition that selects a given set of nurses and weeks to be optimized in each subproblem. Specifically, a subproblem is created by choosing a set of nurses $\mathcal{N}$ that are free to be optimized within a set of weeks $\mathcal{W}$. A nurse that is "free to be optimized" has all variables of shifts and skills unfixed.

In order to explain the selection policy for the elements of $\mathcal{N}$, we first introduce the concepts of block and partition. A *block* is a group composed by $b$ consecutive nurses in an instance. So, $b$ is the block size. A *partition* $P_b$ is a set of blocks of a given size $b$ such that each $n \in N$ appears exactly in one block. For instance, suppose a toy instance with $N = \{1,2,3,4,5,6\}$, a partition composed by blocks of size 2 ($b = 2$) is $P_2 = \{\{1,2\},\{3,4\},\{5,6\}\}$. In the same fashion, if $b = 3$ we have $P_3 = \{\{1,2,3\},\{4,5,6\}\}$. Thus, in a given iteration, $\mathcal{N}$ is defined as a sample of nurses that appears in an element of the sequence of all 2-combinations of $P_b$. Similarly, $\mathcal{W}$ is defined according to the all $k$-combinations of the week set $W$. Since a week have 7 days, $W = \{1, ..., \frac{|D|}{7}\}$. The algorithm starts with a small block size that is increased each time all possibilities of $\mathcal{W}$ are explored. This process is detailed in the next section.

## 4.2 The proposed algorithm

The overall algorithm is described in the pseudo-code of Figure 4.1. Function `LAFO` receives as input a time limit *(TL)* for the algorithm, the time limit for each subproblem *(STL)*, a percentage limit to explore the neighborhood ($\alpha$) and a list size ($\gamma$). The algorithm begins by creating an initial feasible solution $s$ (line 1). We solve the instance considering only the hard constraints of the problem. This approach allows to find an initial feasible solution quickly. If the problem is infeasible, it terminates returning no solution. Line 2 initialize the best solution ($s^*$).

In lines (3–5) a diversification strategy is introduced by a circular list $L_i$, where $i \in 1...\gamma$ are initialized with the objective function value of the initial solution ($Z(s)$). The variables *iter* and $b$ (lines 6–7) count the number of iterations of the main loop and the initial size of blocks, respectively. Line 8 initializes the week set $W$. The outer loop (lines 9–30) iterates until the time limit *(TL)* is reached.

The loop of lines (10–25) iterates over the week set with $k$ being the number o weeks. The next loop iterates over the random sequence of all $k$-combinations of weeks (lines 11–24) by defining $\mathcal{W}$. Thus, for each $\mathcal{W}$, in the loop of lines 12-23, we select $\mathcal{N}$ by iterating in $\alpha$ percent sample of a sequence with all 2-combinations of $P_b$. As a matter of example, suppose we start with 28 days, 4 nurses, and $\alpha$=50%. Thus, we have $b = 1$, $W = \{1,2,3,4\}$, $k = 1$ and $P_1 = \{\{1\},\{2\},\{3\},\{4\}\}$. In the first iteration we could randomly select $\mathcal{W} = \{3\}$ and $\mathcal{N} = \{2,4\}$. In other words, in the subproblem we free the nurses 2 and 4 on all days of the week 3. Since there are six combinations as a result of 2-combinations of $P_1$ and $\alpha$=50%, we iterate more twice in the nurses before changing $\mathcal{W}$. Continuing with the example above, we could randomly select on the second iteration $\mathcal{N} = \{1,2\}$ and in the third iteration $\mathcal{N} = \{1,2\}$, after that we change $\mathcal{W}$.

In line 14 of the algorithm, the implementation of access to the list indexes occurs with the module operators, where $i$ is the list index calculated as iteration $iter$ module of the list size $\gamma$.

In line 13 the subproblem is solved by a MIP model through function `solve()` which receives four parameters: the current solution ($s$), the time limit of the subproblem ($STL$) and the set of variables of nurses and weeks to be optimized ($\mathcal{N}$ and $\mathcal{W}$). This function fixes all $x$ variables to their values in the current solution $s$ and unfixes all variables such as $x_{ndsj} \forall n \in \mathcal{N}, d \in D : d$ corresponds to a week in $\mathcal{W}$, $s \in S, j \in K$. After, the MIP solver is invoked and, if it is able to find a feasible solution within the time limit, it is returned. Otherwise, it returns the previous current solution $s$. Solution $s$ is then stored as the candidate solution $s'$ after the function `solve()`.

In lines 14–21, candidate solution $s'$ is evaluated by the acceptance criterion. A candidate solution is accepted if it is better than the cost of previous iterations kept in $L_i$. When $s'$ is accepted, list $L_i$ and $s$ are updated (lines 16–17). The update of the best solution $s^*$ found is performed only if $Z(s) < Z(s^*)$ (line 18). If so, the new incumbent solution is saved (line 19). After that, the number of iterations is increased (line 22). In line 26, the size of block $b$ is increased. If $b$ is greater than half of the number of nurses, $b$ gets the value 1 (lines 27–29).

Note that since LAFO relies on an existing input MIP model, it also brings the advantages of being easy to maintain and accommodate novel requirements that often arise in a real-world context. Using high-level languages to represent the input MIP Model, as AMPL or MathProg, one could even update the problem requirements without recompiling or retesting the code. Hence, besides keeping the maintenance costs low compared to a standalone heuristic method, our approach provides a more natural and readable way to maintain complex models that evolve with the problem requirements.

Figure 4.1: Pseudocode of the proposed algorithm.

```
Algorithm LAFO (TL, STL, $\alpha$, $\gamma$)
 1: Generate initial feasible solution $s$
 2: $s^* \leftarrow s$
 3: for $i \in 1..\gamma$ do
 4:    $L_i \leftarrow Z(s)$
 5: end for
 6: iter $\leftarrow 0$
 7: $b \leftarrow 1$
 8: $W \leftarrow \{1, ..., \frac{|D|}{7}\}$
 9: repeat
10:    for $k \in 1...|W|$ do
11:       for $\mathcal{W} \in$ random sequence of all k-combinations of $W$ do
12:          for $\mathcal{N} \in \alpha$ percent sample of all 2-combinations of $P_b$ do
13:             $s' \leftarrow$ solve(s, STL, $\mathcal{W}$, $\mathcal{N}$)
14:             i $\leftarrow$ iter mod $\gamma$
15:             if $Z(s') < L_i$ then
16:                $L_i \leftarrow Z(s')$
17:                $s \leftarrow s'$
18:                if $Z(s) < Z(s^*)$ then
19:                   $s^* \leftarrow s$
20:                end if
21:             end if
22:             iter $\leftarrow$ iter $+1$
23:          end for
24:       end for
25:    end for
26:    $b \leftarrow b + 1$
27:    if $b > \frac{|N|}{2}$ then
28:       $b \leftarrow 1$
29:    end if
30: until TL
31: return $s^*$
```

Source: created by author.

# 5 EXPERIMENTAL RESULTS

In this chapter, we present a set of computational experiments designed to help us answer the following questions regarding the NRP problem:

- What requirements make the INRC-II problem difficult to solve with a MIP solver?
- How do the different models $M_1$, $M_2$ and $M_3$ compare to each other?
- What are the best parameters values for the LAFO approach?
- Does the LAFO approach performs better than previous fix-and-optimize methods?
- How does the LAFO approach compares to the state-of-the-art methods?

## 5.1 Experimental setup and data sets

The experimental results were computed in a Desktop-PC equipped with an Intel(R) Xeon(R) CPU E5-2697 (2.70GHz), 64GB of RAM, 24 cores over 64 bits Linux Ubuntu 18.04 operating system. A single core was dedicated to each experiment. All experiments requiring a mathematical model resolution were solved by CPLEX 12.6.0 (IBM, 2015) with default settings using a single core. The algorithms were implemented in $C{+}{+}$ using the compiler $g{+}{+}$[1] (version 7.5.0) with the flags `--std=c++11` and `-O3`. For experiments with LAFO approach we used a seed with value of 1 and the algorithm Mersenne Twister to generate random numbers (MATSUMOTO; NISHIMURA, 1998). All results of the experiments were certified by the INRC-II validator [2] that checks the solution feasibility and also computes the solution value. We have published a repository containing our solutions available at <http://github.com/victoriasimonetti/NurseRostering-INRCII>.

The data sets used in the experiments were the hidden instances. This set comprises all instances with four weeks scheduling horizons provided by the organizers of the INRC-II [3]. These instances are composed of 35, 70 and 110 nurses, four shift types (Early, Day, Late and Night), four skill types (Head Nurse, Nurse, Caretaker and Trainee), and four contract types (FullTime, PartTime, HalfTime and 20Percent).

[1]<https://gcc.gnu.org/>

[2]<http://mobiz.vives.be/inrc2/?page_id=245>

[3]<http://mobiz.vives.be/inrc2/?page_id=20>

Table 5.1 presents the hidden instances tested in this work. The column *Id* identifies an instance of column *Instance*. Also, instances are split into three groups according to the number of nurses, small (35 nurses), medium (70 nurses) and large (110 nurses). The instance names are composed by the number of nurses and weeks, for example, id 1 is the instance n035w4_2_8-8-7-5 that has 35 nurses and 4 weeks.

Table 5.1: Characteristics of the dataset.

| Id | Instance (small) | Id | Instance (medium) | Id | Instance (large) |
|---|---|---|---|---|---|
| 1 | n035w4_2_8-8-7-5 | 11 | n070w4_0_3-6-5-1 | 21 | n110w4_0_1-4-2-8 |
| 2 | n035w4_0_1-7-1-8 | 12 | n070w4_0_4-9-6-7 | 22 | n110w4_0_1-9-3-5 |
| 3 | n035w4_0_4-2-1-6 | 13 | n070w4_0_4-9-7-6 | 23 | n110w4_1_0-1-6-4 |
| 4 | n035w4_0_5-9-5-6 | 14 | n070w4_0_8-6-0-8 | 24 | n110w4_1_0-5-8-8 |
| 5 | n035w4_0_9-8-7-7 | 15 | n070w4_0_9-1-7-5 | 25 | n110w4_1_2-9-2-0 |
| 6 | n035w4_1_0-6-9-2 | 16 | n070w4_1_1-3-8-8 | 26 | n110w4_1_4-8-7-2 |
| 7 | n035w4_2_8-6-7-1 | 17 | n070w4_2_0-5-6-8 | 27 | n110w4_2_0-2-7-0 |
| 8 | n035w4_2_9-2-2-6 | 18 | n070w4_2_3-5-8-2 | 28 | n110w4_2_5-1-3-0 |
| 9 | n035w4_2_9-7-2-2 | 19 | n070w4_2_5-8-2-5 | 29 | n110w4_2_8-9-9-2 |
| 10 | n035w4_2_9-9-2-1 | 20 | n070w4_2_9-5-6-5 | 30 | n110w4_2_9-8-4-9 |

Source: created by author.

## 5.2 Experiments with relaxed versions of the NRP

This section aims to answer which requirements make the INRC-II problem difficult to solve with a MIP solver. This research question was first addressed by Smet (2018) for a correlated nurse rostering problem, and the authors concluded that consecutive requirements influence the most in the computational complexity. To investigate if this conclusion could be supported for the NRP, we evaluate three relaxed problems presented below:

- Relaxed Problem 1 (RP1): NRP without minimum and maximum consecutive requirements. In this experiment, the model $M_0$ was used without *S2b, S2d* and *S3b*.
- Relaxed Problem 2 (RP2): NRP without minimum consecutive requirements. In this experiment, the model $M_0$ was used.
- Relaxed Problem 3 (RP3): NRP without maximum consecutive requirements. In

this experiment, the model $M_3$ was used without *S2b, S2d* and *S3b*.

Table 5.2 reports three different relaxations of the NRP, *RP1*, *RP2* and *RP3*. The column *Id* identifies the instance and the columns *Time* display the time in seconds. Cells marked with "t.l." indicate the time limit of 24 hours was reached without proof of optimality. The objective value of the solution and optimality gap are shown in the columns *Obj* and *Gap*, respectively. The columns *Variables* and *Constraints* present the number of variables and constraints of each model, respectively, after the presolver phase. The last row ($Avg_*$) displays the average values for all instances. Additionally, we also display average values corresponding to small ($Avg_s$), medium ($Avg_m$) and large ($Avg_l$) instances.

Analyzing the results, we can see that RP3 presented the worst performance among the relaxed models. We can note that RP3 was not able to produce optimal results for any instance within the time limit. These results suggest that tackling minimum consecutive requirements significantly affects the performance of the MIP solver. In contrast, in RP1 and RP2, all instances were solved to optimality with an average time of 35 and 181 seconds, respectively. The only exception was the instance 25 in RP1 that reached the time limit with a gap of 2,33%.

If we focus our analysis on RP1 and RP2 according to instance size, we can see that the average time spent on RP2 doubles on small instances compared to RP1. In contrast, this difference is six times greater in medium instances. In general, the time increases when constraints are added, but some instances become particularly hard to solve. For example, instance 15 takes 37.5 seconds to prove optimality in RP1 and approximately 36 minutes on RP2.

As expected, the number of variables and constraints increase according to the size of instances. Also, the difference between the relaxed models affects the number of variables and constraints. When we add maximum constraints (RP2), the average $avg_*$ of variables increases compared with RP1, from $\sim$14 thousand to $\sim$23 thousand. Similarly, this occurs with constraints, from $\sim$7 thousand to $\sim$15 thousand. Moreover, when we add minimum constraints (RP3), the number of variables triples, from $\sim$14 thousand to $\sim$44 thousand, and the number of constraints is five times greater, from $\sim$7 thousand to $\sim$37 thousand.

The results presented here show that minimum consecutive requirements have the most significant impact on the problem computational hardness. Hence, reformulating these constraints could be relevant to produce better results by a MIP model, as shown in the next section.

Table 5.2: Experiment with three relaxations of the NRP: RP1, RP2 and RP3.

| | **RP1: NRP without min. and max. consec.** | | | | | **RP2: NRP without min. consec.** | | | | | **RP3: NRP without max. consec.** | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Id | Time (s) | Obj | Gap | Variables | Constraints | Time (s) | Obj | Gap | Variables | Constraints | Time (s) | Obj | Gap | Variables | Constraints |
| 1 | 117.6 | 70 | 0.0 | 7,218 | 3,532 | 14.4 | 180 | 0.0 | 11,280 | 7,567 | *t.l.* | 300 | 26.7 | 21,469 | 17,920 |
| 2 | 3.0 | 250 | 0.0 | 7,232 | 3,528 | 82.6 | 360 | 0.0 | 11,324 | 7,573 | *t.l.* | 380 | 4.0 | 21,493 | 17,903 |
| 3 | 2.7 | 340 | 0.0 | 7,265 | 3,531 | 33.5 | 490 | 0.0 | 11,308 | 7,539 | *t.l.* | 635 | 5.5 | 21,534 | 17,907 |
| 4 | 2.8 | 280 | 0.0 | 7,114 | 3,519 | 25.6 | 400 | 0.0 | 10,967 | 7,331 | *t.l.* | 660 | 2.8 | 21,428 | 17,892 |
| 5 | 12.3 | 210 | 0.0 | 7,245 | 3,531 | 17.5 | 330 | 0.0 | 11,253 | 7,521 | *t.l.* | 395 | 3.8 | 21,482 | 17,901 |
| 6 | 11.7 | 200 | 0.0 | 7,249 | 3,533 | 56.2 | 330 | 0.0 | 11,217 | 7,458 | *t.l.* | 470 | 6.2 | 21,541 | 17,921 |
| 7 | 2.9 | 220 | 0.0 | 7,257 | 3,537 | 38.7 | 330 | 0.0 | 11,295 | 7,532 | *t.l.* | 430 | 9.7 | 21,526 | 17,925 |
| 8 | 2.9 | 260 | 0.0 | 7,295 | 3,534 | 18.3 | 360 | 0.0 | 11,296 | 7,507 | *t.l.* | 670 | 13.0 | 21,577 | 17,924 |
| 9 | 11.3 | 260 | 0.0 | 7,308 | 3,541 | 29.6 | 390 | 0.0 | 11,324 | 7,533 | *t.l.* | 590 | 7.4 | 21,571 | 17,925 |
| 10 | 5.6 | 280 | 0.0 | 7,252 | 3,536 | 30.9 | 390 | 0.0 | 11,228 | 7,463 | *t.l.* | 565 | 8.4 | 21,551 | 17,920 |
| $Avg_s$ | 17.3 | 237.0 | 0.0 | 7,243.5 | 3,532.2 | 34.7 | 356.0 | 0.0 | 11,249.2 | 7,502.4 | *t.l.* | 509.5 | 8.7 | 21,517.2 | 17,913.8 |
| 11 | 9.1 | 180 | 0.0 | 14,756 | 6,810 | 146.2 | 490 | 0.0 | 22,919 | 14,962 | *t.l.* | 570 | 40.3 | 44,380 | 37,022 |
| 12 | 56.0 | 30 | 0.0 | 14,762 | 6,799 | 109.7 | 280 | 0.0 | 22,955 | 14,972 | *t.l.* | 620 | 75.6 | 44,380 | 37,015 |
| 13 | 224.6 | 30 | 0.0 | 14,772 | 6,809 | 123.3 | 240 | 0.0 | 23,005 | 15,022 | *t.l.* | 835 | 81.2 | 44,380 | 37,015 |
| 14 | 27.6 | 200 | 0.0 | 14,766 | 6,816 | 72.9 | 460 | 0.0 | 22,947 | 14,957 | *t.l.* | 445 | 17.1 | 44,400 | 37,018 |
| 15 | 37.5 | 30 | 0.0 | 14,783 | 6,824 | 2,149.8 | 150 | 0.0 | 23,051 | 15,061 | *t.l.* | 275 | 55.7 | 44,376 | 37,025 |
| 16 | 22.7 | 270 | 0.0 | 14,778 | 6,825 | 93.0 | 500 | 0.0 | 22,876 | 14,902 | *t.l.* | 535 | 17.3 | 44,421 | 37,046 |
| 17 | 31.4 | 130 | 0.0 | 14,783 | 6,822 | 81.8 | 390 | 0.0 | 23,016 | 15,015 | *t.l.* | 520 | 47.6 | 44,419 | 37,036 |
| 18 | 27.5 | 210 | 0.0 | 14,808 | 6,826 | 306.4 | 450 | 0.0 | 23,002 | 14,990 | *t.l.* | 655 | 50.6 | 44,446 | 37,052 |
| 19 | 53.5 | 120 | 0.0 | 14,836 | 6,843 | 113.0 | 400 | 0.0 | 23,099 | 15,066 | *t.l.* | 660 | 53.7 | 44,461 | 37,066 |
| 20 | 31.5 | 30 | 0.0 | 14,787 | 6,823 | 116.0 | 300 | 0.0 | 23,030 | 15,036 | *t.l.* | 720 | 68.3 | 44,403 | 37,037 |
| $Avg_m$ | 52.1 | 123.0 | 0.0 | 14,783.1 | 6,819.7 | 331.2 | 366.0 | 0.0 | 22,990.0 | 14,998.3 | *t.l.* | 583.5 | 50.7 | 44,406.6 | 37,033.2 |
| 21 | 24.2 | 720 | 0.0 | 21,055 | 10,521 | 144.4 | 1,010 | 0.0 | 33,977 | 23,443 | *t.l.* | 865 | 13.3 | 66,162 | 56,468 |
| 22 | 21.2 | 600 | 0.0 | 20,938 | 10,498 | 167.2 | 1,030 | 0.0 | 33,860 | 23,420 | *t.l.* | 815 | 21.5 | 66,045 | 56,445 |
| 23 | 13.8 | 740 | 0.0 | 21,078 | 10,514 | 148.1 | 1,070 | 0.0 | 34,005 | 23,441 | *t.l.* | 800 | 3.8 | 66,167 | 56,443 |
| 24 | 11.0 | 670 | 0.0 | 21,032 | 10,504 | 161.7 | 1,020 | 0.0 | 33,959 | 23,431 | *t.l.* | 740 | 5.4 | 66,121 | 56,433 |
| 25 | *t.l.* | 860 | 2.3 | 21,045 | 10,519 | 255.5 | 1,280 | 0.0 | 33,972 | 23,446 | *t.l.* | 1,030 | 5.8 | 66,131 | 56,445 |
| 26 | 34.0 | 660 | 0.0 | 21,037 | 10,512 | 190.4 | 1,020 | 0.0 | 33,964 | 23,439 | *t.l.* | 740 | 8.1 | 66,126 | 56,441 |
| 27 | 118.8 | 820 | 0.0 | 21,082 | 10,515 | 162.6 | 1,190 | 0.0 | 34,009 | 23,442 | *t.l.* | 920 | 4.4 | 66,171 | 56,444 |
| 28 | 17.3 | 640 | 0.0 | 20,985 | 10,495 | 164.8 | 1,000 | 0.0 | 33,912 | 23,422 | *t.l.* | 780 | 10.3 | 66,071 | 56,421 |
| 29 | 42.6 | 790 | 0.0 | 20,979 | 10,513 | 188.0 | 1,210 | 0.0 | 33,906 | 23,440 | *t.l.* | 955 | 12.6 | 66,068 | 56,442 |
| 30 | 37.2 | 680 | 0.0 | 21,009 | 10,505 | 187.8 | 1,090 | 0.0 | 33,936 | 23,432 | *t.l.* | 875 | 15.8 | 66,095 | 56,431 |
| $Avg_l$ | 35.5 | 718.0 | 0.2 | 21,024.0 | 10,509.6 | 177.0 | 1,092.0 | 0.0 | 33,950.0 | 23,435.6 | *t.l.* | 852.0 | 10.1 | 66,115.7 | 56,441.3 |
| $Avg_*$ | 35.0 | 359.3 | 0.1 | 14,350.2 | 6,953.8 | 181.0 | 604.7 | 0.0 | 22,729.7 | 15,312.1 | *t.l.* | 648.3 | 23.2 | 44,013.2 | 37,129.4 |

## 5.3 Experiments with models $M_1$, $M_2$ and $M_3$

In Section 5.2 we have seen that minimum consecutive requirements seem to be responsible for making the NRP difficult to solve. In this section we evaluate models $M_1$, $M_2$ and $M_3$ that were previously defined in sections 3.2.2, 3.2.3 and 3.2.4. These three models only differ in the way minimum consecutive requirements are formulated.

Table 5.3 displays the comparison results for the models $M_1$, $M_2$ and $M_3$ with a time limit of 24 hours. The column *Id* identifies the instance. Columns labeled *Obj*, *Gap* and *LB* show, respectively, the objective value of the solution, the optimality gap, and the lower bound obtained by the MIP solver to each model. Cells marked with a "-" means that the MIP solver does not found an integer solution. The last row ($Avg_*$) displays the average values for the whole instance group. Additionally, we also display average values corresponding to small ($Avg_s$), medium ($Avg_m$) and, large ($Avg_l$) instances. Best results are shown in bold and running times are suppressed since all runs reached the time limit.

From Table 5.3 we can observe that the model $M_1$ found an integer solution only to the instance 1. For the remaining ones, the MIP solver was not even able to solve the root node due to the high number of variables and constraints required by the consecutive assignment constraints. On the other hand, models $M_2$ and $M_3$ performed better since they found integer solutions for all instances.

When comparing the average results of models $M_1$ and $M_2$ according to the instance size, one can note that while model $M_2$ performed better in large instances, model $M_3$ performed better small and medium instances. On small instances, although the results were very close, with a difference of 4% of the average gap, the model $M_3$ produced the best solutions to 7 out of 10 instances. Concerning medium instances, model $M_3$ also achieved best results than $M_2$ for 8 out of 10 instances with an average gap difference of approximately 22%. In contrast, on large instances, model $M_2$ has achieved the best results for 6 out of 10 instances with an average gap of 8.5% less than model $M_3$. Particularly in large instances, model $M_2$ produced better lower bounds than $M_3$.

These results show empirically how different constraint formulations to the minimum consecutive requirements lead to significant performance differences between the models. While $M_1$ performed poorly, $M_2$ and $M_3$ were able to find feasible solutions to all instances with distinct performance according to the instance size. Since the model $M_3$ presented the best average results we choose it for handling the experiments in the Section 5.2 and with the LAFO approach in the Sections 5.4 and 5.5.

Table 5.3: Comparison between model $M_1$, $M_2$ and $M_3$ with a time limit of 24h.

| | $M_1$ | | | $M_2$ | | | $M_3$ | | |
|---|---|---|---|---|---|---|---|---|---|
| Id | Obj | Gap | LB | Obj | Gap | LB | Obj | Gap | LB |
| 1 | 30,905 | 99.9 | 35 | 1,450 | 49.7 | 730 | **1,355** | 46.0 | 735 |
| 2 | - | - | - | **2,100** | 52.7 | 995 | 2,300 | 57.3 | 985 |
| 3 | - | - | - | 2,140 | 43.8 | 1,205 | **2,110** | 41.8 | 1,230 |
| 4 | - | - | - | **1,770** | 33.8 | 1,175 | 1,800 | 33.0 | 1,190 |
| 5 | - | - | - | 1,790 | 45.1 | 985 | **1,520** | 34.0 | 1,005 |
| 6 | - | - | - | 1,925 | 52.0 | 925 | **1,630** | 42.6 | 935 |
| 7 | - | - | - | 1,540 | 38.9 | 945 | **1,465** | 38.0 | 910 |
| 8 | - | - | - | **2,005** | 44.2 | 1,120 | 2,130 | 46.4 | 1,145 |
| 9 | - | - | - | 1,865 | 41.9 | 1,085 | **1,665** | 33.9 | 1,100 |
| 10 | - | - | - | 1,885 | 44.8 | 1,040 | **1,640** | 33.0 | 1,100 |
| $Avg_s$ | 30,905.0 | 99.9 | 35.0 | 1,847.0 | 44.7 | 1,020.5 | 1,761.5 | 40.7 | 1,033.5 |
| 11 | - | - | - | 4,330 | 63.2 | 1,595 | **3,245** | 42.0 | 1,885 |
| 12 | - | - | - | **3,340** | 59.0 | 1,370 | 3,710 | 54.8 | 1,675 |
| 13 | - | - | - | 5,725 | 75.5 | 1,405 | **3,400** | 51.3 | 1,660 |
| 14 | - | - | - | **2,970** | 44.7 | 1,645 | 3,025 | 37.2 | 1,900 |
| 15 | - | - | - | 12,635 | 89.1 | 1,375 | **2,955** | 45.8 | 1,600 |
| 16 | - | - | - | 6,330 | 72.7 | 1,730 | **3,550** | 44.6 | 1,970 |
| 17 | - | - | - | 5,870 | 73.4 | 1,560 | **2,870** | 38.4 | 1,770 |
| 18 | - | - | - | 3,180 | 46.6 | 1,700 | **3,090** | 40.9 | 1,830 |
| 19 | - | - | - | 3,020 | 46.7 | 1,610 | **2,725** | 32.4 | 1,845 |
| 20 | - | - | - | 10,275 | 84.6 | 1,580 | **3,775** | 52.0 | 1,810 |
| $Avg_m$ | - | - | - | 5,767.5 | 65.6 | 1,557.0 | 3,234.5 | 44.0 | 1,794.5 |
| 21 | - | - | - | **3,660** | 41.4 | 2,150 | 7,515 | 73.8 | 1,965 |
| 22 | - | - | - | **6,545** | 68.7 | 2,050 | 8,105 | 76.6 | 1,895 |
| 23 | - | - | - | **4,955** | 55.9 | 2,185 | 5,285 | 63.3 | 1,940 |
| 24 | - | - | - | 5,100 | 57.1 | 2,190 | **3,330** | 40.4 | 1,985 |
| 25 | - | - | - | **4,045** | 37.5 | 2,530 | 5,875 | 60.8 | 2,305 |
| 26 | - | - | - | **3,080** | 28.8 | 2,195 | 6,820 | 71.3 | 1,960 |
| 27 | - | - | - | 3,710 | 37.2 | 2,330 | **3,595** | 37.9 | 2,230 |
| 28 | - | - | - | 10,305 | 77.8 | 2,290 | **3,555** | 42.7 | 2,040 |
| 29 | - | - | - | **4,250** | 43.5 | 2,405 | 8,770 | 73.9 | 2,290 |
| 30 | - | - | - | 7,575 | 69.0 | 2,350 | **5,700** | 61.6 | 2,190 |
| $Avg_l$ | - | - | - | 5,322.5 | 51.7 | 2,267.5 | 5,855.0 | 60.2 | 2,080.0 |
| $Avg_*$ | 30,905.0 | 99.9 | 35.0 | 4,312.3 | 54.0 | 1,615.0 | 3,617.0 | 48.3 | 1,636.0 |

Source: created by author.

## 5.4 Tuning the parameters of the LAFO algorithm

This section describes the approach we used to set the three main parameters of the LAFO heuristic, namely, the list size ($\gamma$), the subproblem time limit (STL) and the percentage limit to explore the neighborhood ($\alpha$).

Analyzing the performance of the LAFO approach in *ad hoc* experiments, we identify that the parameter $\alpha$ leads to better results when it is defined as a function of the number of nurses ($n$). Therefore, we set $\alpha = \lfloor 8 - 0.06n \rfloor$ in such a way that the percentage limit to explore the neighborhood decreases for larger instances. The remaining parameters, $\gamma$ and STL were set using the irace package in the GNU R software (LÓPEZ-IBÁÑEZ et al., 2016). The irace implements an iterated running procedure that automatically configures application parameters, given an execution budget, a set of instances and a range of suitable values for each parameter. The execution budget is a value that defines the maximum number of executions irace can use to find the best parameters of the input algorithm (LÓPEZ-IBÁNEZ et al., 2016).

For the adjustment experiment, we configured the irace to use instances 1, 4, 8, 11, 14, 18, 21, 24, and 28 as the training set and the remaining ones as the test set. Table 5.4 presents the range of values we used for each parameter. Note that we included a list of size 1 in order to simulate the acceptance criteria of the classical F&O heuristic. Finally, we used a budget of 1,000 with each execution being an average of two runs limited to 5,160 seconds. This time limit was chosen to be in line with the experiments of Section 5.5.

Table 5.4: Parameter ranges evaluated in the irace adjustment procedure.

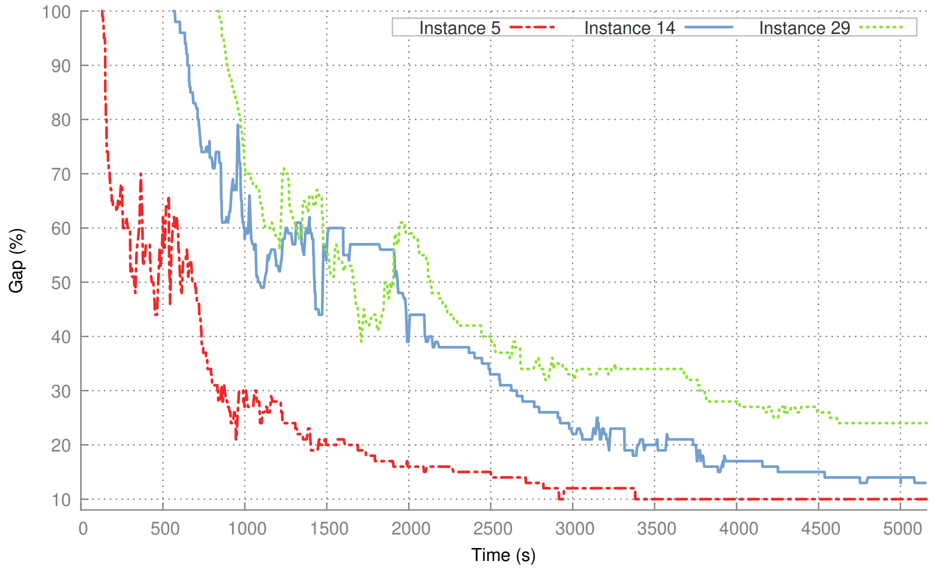
| Parameter | Range of values |
|---|---|
| List size ($\gamma$) | [1,20,50,100,150,200,250,300,500,700] |
| Subproblem time limite (STL) | [1, 5, 8, 10, 15, 30, 60, 120] |

Source: created by author.

The results of irace reported $\gamma = 150$ and $STL = 8$ as the best parameter configuration. Note that, since $STL = 8$ was the same setting found in the work of Wickert, Sartori and Buriol (2016), suggesting it could be an all-rounded value for this parameter. We also can highlight that $\gamma = 1$ was not ranked by irace as a proper setting, this supports the importance of the late acceptance criteria proposed as a diversification mechanism into the fix-and-optimize framework. The LAFO diversification mechanism could be seen in action in Figure 5.1, that present the convergence curve for the current solution gap against time for three instances of different sizes with $\gamma = 150$. One can note that, before achieving convergence, there are significant oscillations in the gap values at the beginning of the search procedure, especially in the first 30 minutes. This behavior was observed in all instances.

The best parameter values reported in this section are used in the LAFO approach in the following experiments.

Figure 5.1: Graphical of a small, medium and large instance with the median result of 10 runs.



Source: created by author.

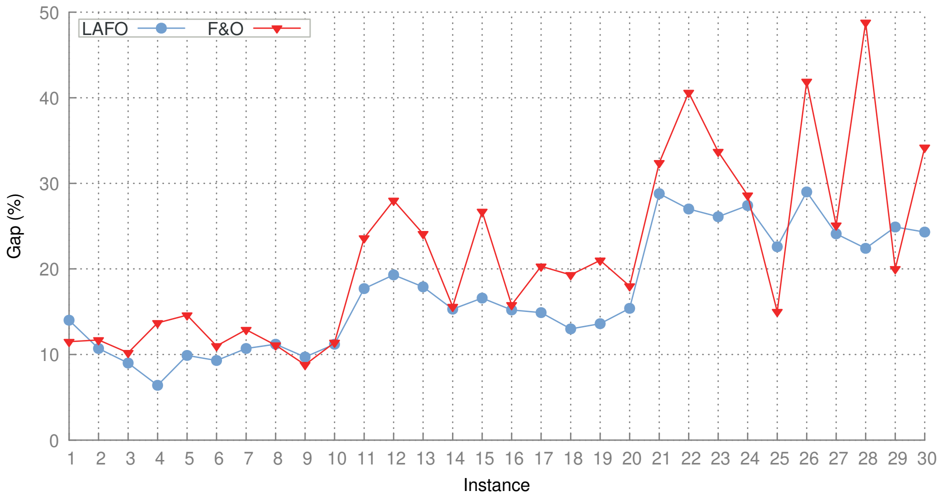
## 5.5 Experiments with LAFO approach and previous F&O methods

In this section, we present the results of the LAFO approach in comparison with the fix-and-optimize (F&O) approach proposed by Wickert, Sartori and Buriol (2016). Our goal is to compare which fix-and-optimize variant performs better to solve the NRP.

In Table 5.5, column *id* identifies the instance. Column *BKS* present the best-known solution values reproduced from different sources. Values marked with a “◇” were obtained by Gomes, Toffolo and Santos (2017), values obtained by Ceschia, Guido and Schaerf (2020) are represented by “§”, and “†” indicates the values obtained by Legrain, Omer and Rosat (2019). Moreover, results marked with “‡” were obtained by Ceschia, Guido and Schaerf (2020) by rerunning some missing experiments of Legrain, Omer and Rosat (2019). While the columns under the label “F&O” show average results for the fix-and-optimize heuristic proposed by Wickert, Sartori and Buriol (2016) with a time limit of

7200 seconds, the columns under the label “LAFO” show average results of our approach evaluated with the model $M_3$. Since the computing facility used by Wickert, Sartori and Buriol (2016)) is slower than ours by a factor of 0.72, we set the time limit of LAFO to be 5160 seconds. This normalization is supported by data from <http://www.cpubenchmark.net/>. For each method, the average objective value($Obj$) for 10 runs is presented with the corresponding average gap ($Gap$) that is calculated as $100 * (\frac{Obj-BKS}{BKS})$. For the LAFO, we report the objective values ($Obj^*$ and gap $Gap^*$) of the best run, as well as the coefficient of variation ($CV$). The row ($Avg_*$) displays the average values for the whole instance group. We also display average values corresponding to small ($Avg_s$), medium ($Avg_m$) and, large ($Avg_l$) instances. The results highlighted in boldface indicate the lowest costs.

Figure 5.2: Graphical comparison between LAFO and F&O.



Source: created by author.

From Table 5.5 we can observe that the proposed algorithm found better solutions to 25 out of 30 instances, as well as an average gap 4.3% better than F&O. If we focus our analysis according to the instance sizes, the LAFO approach achieved the majority of best results in all group sizes, with an average gap difference of 1.5% in small instances, 5.3% in medium instances, and 6.4% in large instances. One can note that the difference between the average gaps obtained by the LAFO and F&O approaches increases along with the instance size. Particularly in large instances, the difference between gaps reached up to $\approx$ 26% for instance 28. Finally, as the average coefficient of variation reported for the LAFO approach is 2%, we consider it a robust algorithm that is able to reproduce solutions with a small deviation among each run. Figure 5.2 presents a graphical comparison between LAFO and F&O.

Table 5.5: Matheuristc LAFO compared with F&O of Wickert, Sartori and Buriol (2016).

| | | F&O | | LAFO | | | | |
|---|---|---|---|---|---|---|---|---|
| Id | BKS | Obj | Gap | $\overline{Obj}$ | $\overline{Gap}$ | Obj$^*$ | Gap$^*$ | CV |
| 1 | 1,085$^\diamond$ | **1,210.0** | 11.5 | 1,237.0 | 14.0 | 1,170 | 7.8 | 3.3 |
| 2 | 1,415$^\dagger$ | 1,580.1 | 11.7 | **1,565.9** | 10.7 | 1,515 | 7.1 | 1.7 |
| 3 | 1,615$^\diamond$ | 1,780.1 | 10.2 | **1,760.5** | 9.0 | 1,705 | 5.6 | 1.1 |
| 4 | 1,530$^*$ | 1,740.2 | 13.7 | **1,628.3** | 6.4 | 1,595 | 4.2 | 0.9 |
| 5 | 1,365$^\diamond$ | 1,565.0 | 14.6 | **1,500.0** | 9.9 | 1,435 | 5.1 | 3.4 |
| 6 | 1,360$^*$ | 1,510.1 | 11.0 | **1,487.0** | 9.3 | 1,465 | 7.7 | 0.6 |
| 7 | 1,315$^*$ | 1,485.0 | 12.9 | **1,455.5** | 10.7 | 1,400 | 6.5 | 2.9 |
| 8 | 1,525$^\diamond$ | **1,695.0** | 11.1 | 1,696.5 | 11.2 | 1,640 | 7.5 | 0.3 |
| 9 | 1,480$^\diamond$ | **1,610.1** | 8.8 | 1,624.0 | 9.7 | 1,570 | 6.1 | 3.0 |
| 10 | 1,485$^*$ | 1,655.0 | 11.4 | **1,651.5** | 11.2 | 1,600 | 7.7 | 1.0 |
| $Avg_s$ | 1,417.5 | 1,583.0 | 11.7 | 1,560.6 | 10.2 | 1,509.5 | 6.5 | 1.8 |
| 11 | 2,415$^*$ | 2,985.1 | 23.6 | **2,842.5** | 17.7 | 2,750 | 13.9 | 1.1 |
| 12 | 2,125$^\dagger$ | 2,719.9 | 28.0 | **2,535.5** | 19.3 | 2,410 | 13.4 | 4.5 |
| 13 | 2,195$^*$ | 2,725.0 | 24.1 | **2,587.0** | 17.9 | 2,515 | 14.6 | 0.6 |
| 14 | 2,315$^*$ | 2,675.1 | 15.6 | **2,668.5** | 15.3 | 2,605 | 12.5 | 2.3 |
| 15 | 2,100$^\ddagger$ | 2,660.1 | 26.7 | **2,448.3** | 16.6 | 2,334 | 11.1 | 1.7 |
| 16 | 2,530$^\ddagger$ | 2,930.0 | 15.8 | **2,915.4** | 15.2 | 2,840 | 12.3 | 3.6 |
| 17 | 2,340$^*$ | 2,815.1 | 20.3 | **2,688.4** | 14.9 | 2,589 | 10.6 | 1.4 |
| 18 | 2,380$^\ddagger$ | 2,839.9 | 19.3 | **2,690.0** | 13.0 | 2,580 | 8.4 | 1.4 |
| 19 | 2,335$^*$ | 2,825.0 | 21.0 | **2,653.4** | 13.6 | 2,565 | 9.9 | 1.0 |
| 20 | 2,395$^*$ | 2,825.0 | 18.0 | **2,764.5** | 15.4 | 2,630 | 9.8 | 1.4 |
| $Avg_m$ | 2,313.0 | 2,800.0 | 21.2 | 2,679.4 | 15.9 | 2,581.8 | 11.6 | 1.9 |
| 21 | 2,345$^*$ | 3,105.1 | 32.4 | **3,020.0** | 28.8 | 2,905 | 23.9 | 4.7 |
| 22 | 2,525$^\dagger$ | 3,550.1 | 40.6 | **3,205.5** | 27.0 | 3,060 | 21.2 | 3.2 |
| 23 | 2,570$^*$ | 3,435.1 | 33.7 | **3,241.0** | 26.1 | 3,115 | 21.2 | 3.0 |
| 24 | 2,555$^*$ | 3,285.0 | 28.6 | **3,254.0** | 27.4 | 3,130 | 22.5 | 2.0 |
| 25 | 2,975$^\ddagger$ | **3,419.9** | 15.0 | 3,646.0 | 22.6 | 3,435 | 15.5 | 1.3 |
| 26 | 2,495$^*$ | 3,539.9 | 41.9 | **3,217.5** | 29.0 | 3,105 | 24.4 | 3.0 |
| 27 | 2,730$^*$ | 3,414.9 | 25.1 | **3,388.5** | 24.1 | 3,260 | 19.4 | 0.9 |
| 28 | 2,685$^*$ | 3,995.0 | 48.8 | **3,285.5** | 22.4 | 3,190 | 18.8 | 1.1 |
| 29 | 2,980$^\ddagger$ | **3,575.2** | 20.0 | 3,720.9 | 24.9 | 3,620 | 21.5 | 2.8 |
| 30 | 2,775$^\ddagger$ | 3,724.4 | 34.2 | **3,449.0** | 24.3 | 3,335 | 20.2 | 0.5 |
| $Avg_l$ | 2,663.5 | 3,504.5 | 32.0 | 3,342.8 | 25.6 | 3,215.5 | 20.9 | 2.2 |
| $Avg_*$ | 2,131.3 | 2,629.2 | 21.7 | 2,527.6 | 17.3 | 2,435.6 | 13.0 | 2.0 |

Source: created by author.

## 5.6 Comparison between LAFO and the state-of-the-art results from the literature

Table 5.6 presents a comparison between our approach and the the state-of-the-art methods for the NRP. Column *BKS* reproduces the best known solutions presented in Table 5.5. The following columns displays, for each method, a objective value (*Obj*), a gap (*Gap*), that is calculated using the equation $gap = 100 * (\frac{Obj-BKS}{BKS})$. The time in seconds (*Time (s)*) was normalized to machine used in the paper of Wickert, Sartori and Buriol (2016) (see Section 5.5), except the work of Gomes et al. (2017) because the algorithm was executed in parallel. Results of Ceschia, Guido and Schaerf (2020), and the LAFO approach were reported as an average of 10 runs. The results highlighted in boldface indicate the lowest costs.

From Table 5.6 we can observe that the best solutions were found in different instances by Legrain, Omer and Rosat (2019), Gomes, Toffolo and Santos (2017), and Ceschia, Guido and Schaerf (2020). However, the time limit used by Gomes et al. (2017) is significantly greater when compared with the others methods, even not normalized. The best average gap in small instances was obtained by Gomes et al. (2017) with 0.5%, and Legrain et al. (2019) achieved the best average gap on medium and large instances with 0.6%, and 1.4%, respectively. The LAFO approach obtained gaps significantly greater than the others methods, specially in medium and large instances. This result suggests that the fix-and-optimize, overall, still has issues that need to be tackled to be competitive with state-of-the-art methods. Figure 5.3 presents a graphical comparison between LAFO and the state-of-the-art results from the literature.

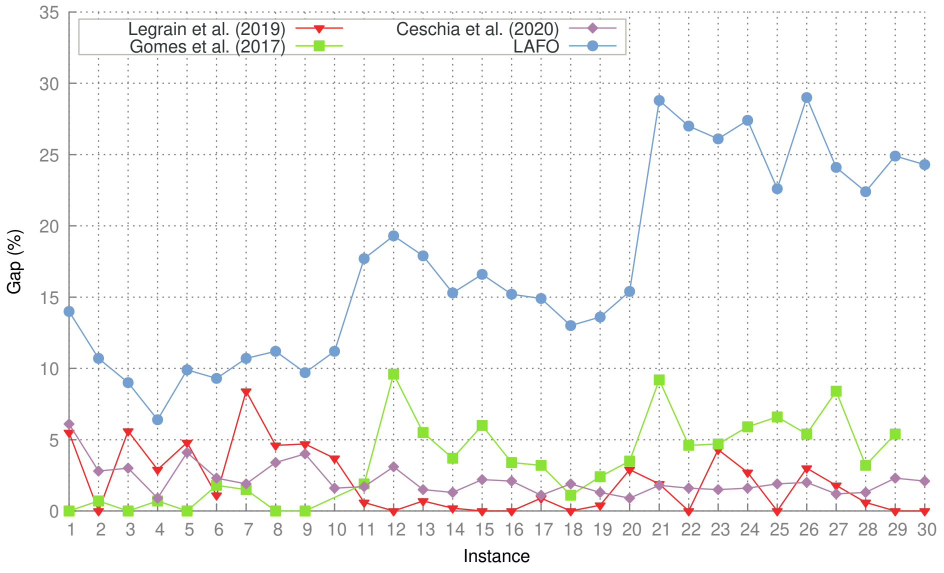
Table 5.6: Comparison between LAFO and the state-of-the-art results from the literature.

| | | **Legrain et al. (2019)** | | | **Gomes et al. (2017)** | | | **Ceschia et al. (2020)** | | | **LAFO** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Id | BKS | Obj | Gap | Time (s) | Obj | Gap | Time (s)* | Obj | Gap | Time (s) | Obj | Gap | Time (s) |
| 1 | 1,085 | 1,145 | 5.5 | 1,803 | **1,085** | 0.0 | 5,586 | 1,151.0 | 6.1 | 1,317 | 1,237.0 | 14.0 | 5,160 |
| 2 | 1,415 | **1,415** | 0.0 | 1,803 | 1,425 | 0.7 | 3,269 | 1,455.0 | 2.8 | 1,317 | 1,565.9 | 10.7 | 5,160 |
| 3 | 1,615 | 1,705 | 5.6 | 1,803 | **1,615** | 0.0 | 5,124 | 1,663.0 | 3.0 | 1,317 | 1,760.5 | 9.0 | 5,160 |
| 4 | 1,530 | 1,575 | 2.9 | 1,803 | **1,540** | 0.7 | 6,872 | 1,544.5 | 0.9 | 1,317 | 1,628.3 | 6.4 | 5,160 |
| 5 | 1,365 | 1,430 | 4.8 | 1,803 | **1,365** | 0.0 | 4,475 | 1,421.5 | 4.1 | 1,317 | 1,500.0 | 9.9 | 5,160 |
| 6 | 1,360 | **1,375** | 1.1 | 1,803 | 1,385 | 1.8 | 5,359 | 1,391.5 | 2.3 | 1,317 | 1,487.0 | 9.3 | 5,160 |
| 7 | 1,315 | 1,425 | 8.4 | 1,803 | **1,335** | 1.5 | 6,453 | 1,340.5 | 1.9 | 1,317 | 1,455.5 | 10.7 | 5,160 |
| 8 | 1,525 | 1,595 | 4.6 | 1,803 | **1,525** | 0.0 | 6,204 | 1,577.5 | 3.4 | 1,317 | 1,696.5 | 11.2 | 5,160 |
| 9 | 1,480 | 1,550 | 4.7 | 1,803 | **1,480** | 0.0 | 12,340 | 1,539.5 | 4.0 | 1,317 | 1,624.0 | 9.7 | 5,160 |
| 10 | 1,485 | 1,540 | 3.7 | 1,803 | – | – | – | **1,509.0** | 1.6 | 1,317 | 1,651.5 | 11.2 | 5,160 |
| $Avg_s$ | 1,417.5 | 1,475.5 | 4.1 | 1,803.0 | 1,417 | 0.5 | 6,186.9 | 1,459.3 | 3.0 | 1,317.0 | 1,560.6 | 10.2 | 5,160.0 |
| 11 | 2,415 | **2,430** | 0.6 | 3,206 | 2,460 | 1.9 | 3,640 | 2,455.0 | 1.7 | 2,342 | 2,842.5 | 17.7 | 5,160 |
| 12 | 2,125 | **2,125** | 0.0 | 3,206 | 2,330 | 9.6 | 4,943 | 2,190.0 | 3.1 | 2,342 | 2,535.5 | 19.3 | 5,160 |
| 13 | 2,195 | **2,210** | 0.7 | 3,206 | 2,315 | 5.5 | 9,465 | 2,229.0 | 1.5 | 2,342 | 2,587.0 | 17.9 | 5,160 |
| 14 | 2,315 | **2,320** | 0.2 | 3,206 | 2,400 | 3.7 | 1,795 | 2,345.5 | 1.3 | 2,342 | 2,668.5 | 15.3 | 5,160 |
| 15 | 2,100 | **2,100** | 0.0 | 2,342 | 2,225 | 6.0 | 3,395 | 2,147.0 | 2.2 | 2,342 | 2,448.3 | 16.6 | 5,160 |
| 16 | 2,530 | **2,530** | 0.0 | 2,342 | 2,615 | 3.4 | 3,457 | 2,582.5 | 2.1 | 2,342 | 2,915.4 | 15.2 | 5,160 |
| 17 | 2,340 | 2,360 | 0.9 | 3,206 | 2,415 | 3.2 | 2,990 | **2,365.0** | 1.1 | 2,342 | 2,688.4 | 14.9 | 5,160 |
| 18 | 2,380 | **2,380** | 0.0 | 2,342 | 2,405 | 1.1 | 5,032 | 2,424.5 | 1.9 | 2,342 | 2,690.0 | 13.0 | 5,160 |
| 19 | 2,335 | **2,345** | 0.4 | 3,206 | 2,390 | 2.4 | 7,580 | 2,366.5 | 1.3 | 2,342 | 2,653.4 | 13.6 | 5,160 |
| 20 | 2,395 | 2,465 | 2.9 | 3,206 | 2,480 | 3.5 | 2,495 | **2,416.0** | 0.9 | 2,342 | 2,764.5 | 15.4 | 5,160 |
| $Avg_m$ | 2,313.0 | 2,326.5 | 0.6 | 2,946.0 | 2,404 | 4.0 | 4,479.2 | 2,352.1 | 1.7 | 2,342.0 | 2,679.4 | 15.9 | 5,160.0 |
| 21 | 2,345 | 2,390 | 1.9 | 4,809 | 2,560 | 9.2 | 13,084 | **2,387.5** | 1.8 | 3,513 | 3,020.0 | 28.8 | 5,160 |
| 22 | 2,525 | **2,525** | 0.0 | 4,809 | 2,640 | 4.6 | 9,624 | 2,566.5 | 1.6 | 3,513 | 3,205.5 | 27.0 | 5,160 |
| 23 | 2,570 | 2,680 | 4.3 | 4,809 | 2,690 | 4.7 | 24,585 | **2,609.0** | 1.5 | 3,513 | 3,241.0 | 26.1 | 5,160 |
| 24 | 2,555 | 2,625 | 2.7 | 4,809 | 2,705 | 5.9 | 12,838 | **2,596.0** | 1.6 | 3,513 | 3,254.0 | 27.4 | 5,160 |
| 25 | 2,975 | **2,975** | 0.0 | 3,513 | 3,170 | 6.6 | 11,570 | 3,032.0 | 1.9 | 3,513 | 3,646.0 | 22.6 | 5,160 |
| 26 | 2,495 | 2,570 | 3.0 | 4,809 | 2,630 | 5.4 | 8,350 | **2,545.5** | 2.0 | 3,513 | 3,217.5 | 29.0 | 5,160 |
| 27 | 2,730 | 2,780 | 1.8 | 4,809 | 2,960 | 8.4 | 10,882 | **2,763.5** | 1.2 | 3,513 | 3,388.5 | 24.1 | 5,160 |
| 28 | 2,685 | **2,700** | 0.6 | 4,809 | 2,770 | 3.2 | 9,079 | 2,719.0 | 1.3 | 3,513 | 3,285.5 | 22.4 | 5,160 |
| 29 | 2,980 | **2,980** | 0.0 | 3,513 | 3,140 | 5.4 | 15,184 | 3,049.0 | 2.3 | 3,513 | 3,720.9 | 24.9 | 5,160 |
| 30 | 2,775 | **2,775** | 0.0 | 3,513 | 3,005 | 8.3 | 11,311 | 2,834.0 | 2.1 | 3,513 | 3,449.0 | 24.3 | 5,160 |
| $Avg_l$ | 2663.5 | 2700.0 | 1.4 | 4,420.0 | 2,827.0 | 6.1 | 12650.7 | 2,710.2 | 1.7 | 3,513.0 | 3,342.8 | 25.6 | 5160.0 |
| $Avg_*$ | 2,131.3 | 2,167.3 | 2.0 | 3,056.0 | 2,215.9 | 3.6 | 7,772.3 | 2,173.9 | 2.2 | 2,390.0 | 2,527.6 | 17.3 | 5,160.0 |

Source: created by author.

* Time limit of Gomes et al. (2017) are not normalized because the algorithm was executed in parallel.

Figure 5.3: Graphical comparison between LAFO and the state-of-the-art results.



Source: created by author.

# 6 CONCLUSIONS

The research carried out in this work presents a study about the static version of the INRC-II problem. Different from the majority of the works proposed in the related literature, in this research, we investigated the limitations of MIP approaches, including how different requirements impact the resolution of the problem and how this approach could perform when inserted in a matheuristic procedure.

In addition to defining the NRP, MIP models were proposed to study the problem and evaluate its performance. The experimental results revealed that minimum consecutive requirements have the most significant impact on the problem in the resolution time. When added to the model, it was not able to produce optimal results for any instance within the time limit of 24 hours. For this purpose, we reformulated these requirements that resulted in three different models. The model $M_1$ has a formulation that consecutive assignments are identified as blocks. The model $M_2$ identify patterns for all sizes of minimum consecutive requirements with one set of constraint and using a Big-M. In the model $M_3$, each constraint identifies one pattern without using a Big-M. The experimental results performed on INRC-II instances revealed that model $M_3$ produces better solutions than the other ones when solved by a MIP solver. However, the obtained results demonstrated that solving the NRP problem with a MIP model is still inefficient even when a time limit of 24 hours is available.

In addition to mathematical models, a novel approach was proposed for solving the NRP by exploring weeks and nurses decompositions into blocks through a fix-and-optimize heuristic combined with Late Acceptance criteria. An initial experimental investigation for setting the parameters of the LAFO approach demonstrated that, while the $\alpha$ parameter led to better results when it was defined as a function of the number of nurses, the remaining parameters were better when adjusted by irace. One of these values reported by irace was $STL$ with 8 seconds, the same setting found in the work of Wickert, Sartori and Buriol (2016), suggesting it could be an all-rounded value for this parameter. Also, the value defined to $\gamma = 150$ supports the importance of the LA criteria proposed as a diversification mechanism into the F&O. When compared with a previously reported fix-and-optimize approach proposed by Wickert, Sartori and Buriol (2016), the results demonstrated that the LAFO approach performs, on average, 4.4% better for solving the NRP.

When compared with state-of-the-art approaches, the LAFO approach provides solutions that are about 15% far from the best method, proposed by Legrain, Omer and Rosat (2018) in terms of quality. Although, a proper time comparison between methods was not possible because the approach of Gomes, Toffolo and Santos (2017) was executed in parallel. Finally, the LAFO approach is easy to maintain and accommodate novel requirements by only changing the corresponding MIP models, i.e., it could provide an interesting trade-off between solution quality and software maintenance, at the same time that allows scaling large instances better than a general purpose MIP solver.

As future work, there are several directions in which the research conducted in this work can be extended: (i) explore different strategies for selecting partitions in the fix-and-optimize heuristic (ii) investigate algorithms to generate initial solutions (iii) network flow model with attention to the consecutiveness constraints.

## REFERENCES

ARENDT, J. Shift work: coping with the biological clock. **Occupational medicine**, Oxford University Press, v. 60, n. 1, p. 10–20, 2010.

ASTA, S.; ÖZCAN, E.; CURTOIS, T. A tensor based hyper-heuristic for nurse rostering. **Knowledge-based systems**, Elsevier, v. 98, p. 185–199, 2016.

BERGH, J. Van den et al. Personnel scheduling: A literature review. **European Journal of Operational Research**, Elsevier, v. 226, n. 3, p. 367–385, 2013.

BERRADA, I.; FERLAND, J. A.; MICHELON, P. A multi-objective approach to nurse scheduling with both hard and soft constraints. **Socio-economic planning sciences**, Pergamon, v. 30, n. 3, p. 183–193, 1996.

BILGIN, B. et al. A hyper-heuristic combined with a greedy shuffle approach to the nurse rostering competition. In: **Proceedings of the 8th International Conference on the Practice and Theory of Automated Timetabling (PATAT–2010)**. [S.l.: s.n.], 2010.

BURKE, E. K.; BYKOV, Y. A late acceptance strategy in hill-climbing for exam timetabling problems. In: **Proceedings of the 7th International Conference on the Practice and Theory of Automated Timetabling (PATAT–2008)**. [S.l.: s.n.], 2008.

BURKE, E. K. et al. The state of the art of nurse rostering. **Journal of scheduling**, Kluwer Academic Publishers, v. 7, n. 6, p. 441–499, 2004.

BURKE, E. K.; CURTOIS, T. An ejection chain method and a branch and price algorithm applied to the instances of the first international nurse rostering competition, 2010. In: **Proceedings of the 8th International Conference on the Practice and Theory of Automated Timetabling (PATAT–2010)**. [S.l.: s.n.], 2010. v. 10, p. 13.

BURKE, E. K.; CURTOIS, T. New approaches to nurse rostering benchmark instances. **European Journal of Operational Research**, Elsevier, v. 237, n. 1, p. 71–81, 2014.

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**, Elsevier, v. 203, n. 2, p. 484–493, 2010.

CAUSMAECKER, P. D. et al. Analysis of real-world personnel scheduling problems. In: **Proceedings of the 5th International Conference on the Practice and Theory of Automated Timetabling (PATAT–2004)**. Pittsburgh, USA: [s.n.], 2004. p. 183–197.

CESCHIA, S. et al. The second international nurse rostering competition. **Annals of Operations Research**, Springer, v. 274, n. 1, p. 171–186, 2019.

CESCHIA, S.; GUIDO, R.; SCHAERF, A. Solving the static inrc-ii nurse rostering problem by simulated annealing based on large neighborhoods. **Annals of Operations Research**, Springer, p. 1–19, 2020.

COSTA, E. d. S.; MORITA, I.; MARTINEZ, M. A. Percepção dos efeitos do trabalho em turnos sobre a saúde e a vida social em funcionários da enfermagem em um hospital universitário do estado de são paulo. **Cad Saúde Pública**, SciELO Brasil, v. 16, n. 2, p. 553–5, 2000.

DORNELES, Á. P.; ARAÚJO, O.; BURIOL, L. The impact of compactness requirements on the resolution of high school timetabling problem. In: **Congreso Latino-Iberoamericano de Investigación Operativa**. [S.l.: s.n.], 2012.

GINTNER, V.; KLIEWER, N.; SUHL, L. Solving large multiple-depot multiple-vehicle-type bus scheduling problems in practice. **OR Spectrum**, Springer, v. 27, n. 4, p. 507–523, 2005.

GOMES, R. A.; TOFFOLO, T. A.; SANTOS, H. G. Variable neighborhood search accelerated column generation for the nurse rostering problem. **Electronic Notes in Discrete Mathematics**, Elsevier, v. 58, p. 31–38, 2017.

HASPESLAGH, S. et al. First international nurse rostering competition 2010 (august 10-13, 2010, belfast, uk). In: **Proceedings of the 8th International Conference on the Practice and Theory of Automated Timetabling (PATAT–2010)**. [S.l.: s.n.], 2010.

ISKEN, M.; HANCOCK, W. A heuristic approach to nurse scheduling in hospital units with non-stationary, urgent demand, and a fixed staff size. **Journal of the Society for Health Systems**, v. 2, n. 2, p. 24–41, 1991.

JIN, H.; POST, G.; VEEN, E. van der. Ortecs contribution to the second international nurse rostering competition. In: **Proceedings of the 11th International Conference on the Practice and Theory of Automated Timetabling (PATAT–2016)**. Udine, Italy: [s.n.], 2016. p. 499–501.

KHEIRI, A. et al. A sequence-based selection hyper-heuristic: Case study in multi-stage nurse rostering problem. In: **Proceedings of the 11th International Conference on the Practice andTtheory of Automated Timetabling (PATAT–2016)**. Udine, Italy: [s.n.], 2016. p. 503–505.

LEGRAIN, A.; OMER, J.; ROSAT, S. An online stochastic algorithm for a dynamic nurse scheduling problem. **European Journal of Operational Research**, Elsevier, 2018.

LEGRAIN, A.; OMER, J.; ROSAT, S. A rotation-based branch-and-price approach for the nurse scheduling problem. **Mathematical Programming Computation**, Springer, p. 1–34, 2019.

LÓPEZ-IBÁÑEZ, M. et al. The irace package: Iterated racing for automatic algorithm configuration. **Operations Research Perspectives**, Elsevier, v. 3, p. 43–58, 2016.

LÓPEZ-IBÁNEZ, M. et al. The irace package: Iterated racing for automatic algorithm configuration. **Operations Research Perspectives**, Elsevier, v. 3, p. 43–58, 2016. ISSN 2214-7160.

MASON, A. J.; SMITH, M. C. A nested column generator for solving rostering problems with integer programming. In: CURTIN UNIVERSITY OF TECHNOLOGY PERTH, AUSTRALIA. **International conference on optimisation: techniques and applications**. [S.l.], 1998. p. 827–834.

MATSUMOTO, M.; NISHIMURA, T. Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. **ACM Transactions on Modeling and Computer Simulation (TOMACS)**, ACM, v. 8, n. 1, p. 3–30, 1998.

MISCHEK, F.; MUSLIU, N. Integer programming model extensions for a multi-stage nurse rostering problem. **Annals of operations research**, Springer, v. 275, n. 1, p. 123–143, 2019.

OSOGAMI, T.; IMAI, H. Classification of various neighborhood operations for the nurse scheduling problem. **Algorithms and computation**, Springer, p. 72–83, 2000.

POCHET, Y.; WOLSEY, L. A. **Production planning by mixed integer programming**. New York, USA: Springer Science & Business Media, 2006. 77–113 p.

RÖMER, M.; MELLOULI, T. 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 (PATAT–2016)**. Udine, Italy: [s.n.], 2016. p. 549–551.

RÖMER, M.; MELLOULI, T. Future demand uncertainty in personnel scheduling: investigating deterministic lookahead policies using optimization and simulation. In: **Proceedings of the 30th European conference on modelling and simulation (ECMS, 2016)**. [S.l.: s.n.], 2016. p. 502–507.

SANTOS, H. G. et al. Integer programming techniques for the nurse rostering problem. **Annals of Operations Research**, Springer, v. 239, n. 1, p. 225–251, 2014.

SMET, P. Constraint reformulation for nurse rostering problems. In: PATAT. **Proceedings of the 12th international conference on the practice and theory of automated timetabling**. [S.l.], 2018. p. 69–80.

VALOUXIS, C. et al. A systematic two phase approach for the nurse rostering problem. **European Journal of Operational Research**, Elsevier, v. 219, n. 2, p. 425–433, 2012.

WARNER, D. M.; PRAWDA, J. A mathematical programming model for scheduling nursing personnel in a hospital. **Management Science**, INFORMS, v. 19, n. 4-part-1, p. 411–422, 1972.

WICKERT, T.; SARTORI, C.; BURIOL, L. A fix-and-optimize vns algorithm applied to the nurse rostering problem. In: **Proceedings of the Sixth International Workshop on Model-based Metaheuristic (Matheuristics–2016)**. [S.l.: s.n.], 2016. p. 1–12.

WICKERT, T. I. **Personnel rostering: models and algorithms for scheduling, rescheduling and ensuring robustness**. Thesis (PhD), 2019.

WICKERT, T. I.; SMET, P.; BERGHE, G. V. The nurse rerostering problem: Strategies for reconstructing disrupted schedules. **Computers & Operations Research**, Elsevier, v. 104, p. 319–337, 2019.

WOLFE, H.; YOUNG, J. P. Staffing the nursing unit: Part i. controlled variable staffing. **Nursing Research**, LWW, v. 14, n. 3, p. 236–242, 1965.

WOLFE, H.; YOUNG, J. P. Staffing the nursing unit part ii. the multiple assignment technique. **Nursing Research**, LWW, v. 14, n. 4, p. 299–303, 1965.

WOLSEY, L. A. **Integer programming**. London: Wiley, 1998. 288 p.

# APPENDIX A — RESUMO EXPANDIDO

## Modelos Matemáticos e uma *Matheuristic Late Acceptance Fix-and-Otimize* para um problema de escalonamento de enfermeiros

A construção de escalas de trabalho de enfermeiros é um processo operacional presente em setores hospitalares que prestam serviços de enfermagem 24 horas por dia e 7 dias por semana. Esse processo, denominado Problema de Escalonamento de Enfermeiros (PEE), consiste em designar um determinado conjunto de enfermeiros para turnos de trabalho distribuídos em um horizonte de planejamento de várias semanas. Além de construir uma escala viável, uma solução para o PEE precisa considerar diversos requisitos, como leis trabalhistas, normas institucionais e preferências dos funcionários. O alto número de requisitos associados à natureza combinatória do problema resulta em um processo que pode levar vários dias para ser resolvido manualmente, e ainda assim produzir uma escala de baixa qualidade.

A construção de uma escala influencia diretamente na qualidade da assistência do serviço de enfermagem. Uma escala ruim pode resultar em várias consequências negativas, tais como: (i) excesso de trabalho, (ii) ausências injustificadas, (iii) doença, (iv) conflitos internos entre os membros da equipe, e (v) erros nos procedimentos de cuidados de enfermagem que põe em perigo a vida do paciente. Assim, a automatização dessa tarefa por softwares torna-se imprescindível para garantir a qualidade do serviço, principalmente em hospitais de médio e grande porte (COSTA; MORITA; MARTINEZ, 2000; ARENDT, 2010).

Além de ser um problema de otimização NP-Hard (OSOGAMI; IMAI, 2000), a pesquisa sobre o PEE ainda é desafiadora devido às diversas variações do problema propostas na literatura que possuem diferentes objetivos e requisitos. Por esse motivo torna-se difícil a comparação do desempenho de diferentes métodos de resolução. Para solucionar essa questão foram organizadas competições com o objetivo de comparar técnicas e estimular a publicação de resultados em versões padronizadas do PEE. A versão mais recente do problema foi formalizada na *Second International Nurse Rostering Competition* (INRC-II) (CESCHIA et al., 2019). Nesta competição, foi proposto um problema *multi-estágio*, onde soluções para uma semana deveriam ser produzidas sequencialmente, sem informação prévia sobre os requisitos nas semanas seguintes. Enquanto a versão de múltiplos estágios é útil em uma aplicação prática para produzir soluções ótimas para o INRC-II e avaliar a qualidade das abordagens de previsão, os pesquisadores usaram a chamada versão *estática* do problema INRC-II. Na versão estática, em vez de resolver cada semana sequencialmente, todas são resolvidas de uma só vez.

Para a versão estática do INRC-II, técnicas avançadas de resolução foram desenvolvidas, incluindo Programação Inteira (IP) e metaheurísticas (WICKERT; SARTORI; BURIOL, 2016; GOMES; TOFFOLO; SANTOS, 2017; LEGRAIN; OMER; ROSAT, 2019; CESCHIA; GUIDO; SCHAERF, 2020). É importante ressaltar que, embora haja um entendimento comum entre os pesquisadores de que é difícil resolver a versão estática por meio de uma abordagem MIP, até onde sabemos, não há publicação com resultados numéricos que apoiem essa afirmação. Essa lacuna motiva a investigação das limitações das abordagens MIP, incluindo como diferentes requisitos impactam na resolução do problema e como essa abordagem poderia funcionar quando inserida em um procedimento matemático.

Neste trabalho, focamos na versão *estática* do problema INRC-II. Além da definição do PEE, foram propostos modelos MIP para estudar o problema e avaliar seu desempenho. Os resultados experimentais revelaram que os requisitos mínimos consecutivos têm o impacto mais significativo no tempo de resolução. Quando adicionados ao modelo, o MIP não foi capaz de produzir resultados ótimos para nenhuma instância dentro do limite de tempo de 24 horas. Dessa forma, reformulamos esses requisitos que resultaram em três modelos distintos. O modelo $M_1$ tem uma formulação em que atribuições consecutivas são identificadas como blocos. O modelo $M_2$ identifica padrões para todos os tamanhos de requisitos mínimos consecutivos com um conjunto de restrições e usando um Big-M. No modelo $M_3$, cada restrição identifica um padrão sem usar um Big-M. Os resultados experimentais realizados nas instâncias da INRC-II revelaram que o modelo $M_3$ produz soluções melhores do que os outros quando resolvido por um solver MIP. No entanto, os resultados obtidos demonstraram que resolver o problema de PEE com um modelo MIP ainda é ineficiente mesmo quando o limite de tempo de 24 horas está disponível.

Além dos modelos matemáticos, uma nova abordagem foi proposta para resolver o PEE, explorando semanas e decomposições de enfermeiros em blocos por meio de uma heurística *fix-and-optimize* combinada com critérios de Aceitação Tardia, do inglês *Late Acceptance Fix-and-Optimize* (LAFO). Uma investigação experimental inicial para definir os parâmetros da abordagem LAFO demonstrou que, enquanto o parâmetro $alpha$ levou a melhores resultados quando foi definido em função do número de enfermeiros, os demais parâmetros foram melhores quando ajustados pelo irace. Um desses valores reportados pela irace foi $STL$ com 8 segundos, a mesma configuração encontrada no trabalho de Wickert, Sartori and Buriol (2016), sugerindo que poderia ser um valor ideal para este parâmetro. Além disso, o valor definido para $\gamma = 150$ apoia a importância dos critérios de aceitação tardia propostos como um mecanismo de diversificação para o *fix-and-optimize*. Quando comparado com uma abordagem de *fix-and-optimize* estudada anteriormente por Wickert, Sartori and Buriol (2016), os resultados demonstraram que a abordagem LAFO tem desempenho, em média, 4,4% melhor para resolver o PEE.

Quando comparada ao estado da arte, a abordagem LAFO fornece soluções que estão cerca de 15% longe do melhor método proposto por Legrain, Omer and Rosat (2018) em termos de qualidade. Porém, uma comparação adequada de tempos entre os métodos não foi possível porque a abordagem da Gomes, Toffolo and Santos (2017) foi executada em paralelo. Finalmente, a abordagem LAFO é fácil de manter e adicionar novos requisitos, alterando apenas os modelos de MIP correspondentes, ou seja, pode fornecer um equilíbrio interessante entre qualidade da solução e manutenção de software, ao mesmo tempo que permite escalar melhor grandes instâncias quando comparado ao modelo matemático proposto.