Expert Systems With Applications 112 (2018) 62–76

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Integrated staff scheduling at a medical emergency service: An optimisation approach

Hendrik Vermuyten[a,b,*], Joana Namorado Rosa[a], Inês Marques[a], Jeroen Beliën[b,c], Ana Barbosa-Póvoa[a]

[a] *Centre for Management Studies, Instituto Superior Técnico, University of Lisbon, Av. Rovisco Pais, 1, Lisbon, 1049-001 Portugal*
[b] *Faculty of Economics and Business, Department of Information Management, Modeling and Simulation, KU Leuven Campus Brussels, Warmoesberg, 26, Brussels, 1000 Belgium*
[c] *Faculty of Economics and Business, Department of Decision Sciences and Information Management, KU Leuven, Naamsestraat 69, Leuven, 3000 Belgium*

ARTICLE INFO

Article history:
Received 28 September 2017
Revised 7 December 2017
Accepted 7 June 2018
Available online 15 June 2018

Keywords:
Staff scheduling
Emergency medical services
Diving heuristic
Column generation
Variable neighbourhood decomposition search
Fix-and-optimise

ABSTRACT

Personnel scheduling is a difficult problem with many specific requirements that differ between industries or companies. This paper proposes an Integer Programming (IP) formulation for the staff scheduling problem encountered in practice at an Emergency Medical Services (EMS) system. Two types of heuristics (a diving heuristic and a VNDS heuristic) are implemented and extensively tested on a set of problem instances with different dimensions. Results show that the VNDS heuristic clearly outperforms both the diving heuristic and a state-of-the-art commercial IP solver. It is able to find good quality solutions for realistic problem instances in relatively short computation times. The characteristics that determine the relative difficulty of a problem instance are also investigated. Furthermore, the model is applied to a case study at the Portuguese National Institute for Medical Emergencies. For this purpose, the VNDS heuristic has been implemented in an expert system with an easy-to-use graphical user interface. A number of different schedules proposed by the system are compared with the schedule implemented in practice. This analysis shows that the VNDS heuristic is a significant improvement over the current manual scheduling procedure. Moreover, two what-if scenarios are described to show how the expert system can be used to assist managers in making decisions on contracting additional staff or providing training to workers.

1. Introduction

Staff scheduling is a common problem to most organisations and has been widely studied in the literature. Ernst, Jiang, Krishnamoorthy, and Sier (2004b) define staff scheduling as the crucial process of deploying timetables for a set of workers within an organisation to sat isfy demands for various services, while simultaneously ensuring a distinctive level of employee satisfaction. Moreover, models also need to consider legal, organisational and contractual constraints (Vanden Berghe, 2013).

In Emergency Medical Services (EMS) systems, staff scheduling is of paramount importance, since shortages in the number of require d personnel directly impact the quality of care patients receive. Additionally, the shift sequences assigned to personnel also impact their competency. If someone is, e.g., assigned to a morning shift after doing a night shift the previous day, this can lead to so-called jet fatigue (Kreeft, 2012). Furthermore, employee satisfaction cannot be neglected, as undesirable schedules can lead to increased staff turnover (Cline, Reilly, & Moore, 2003). In fact, fulfilment of employee preferences is directly related to on-job-performance and for this reason the relative attention given to this aspect in scheduling decisions has grown (Vanden Berghe, 2013).

This research was motivated by the staff scheduling problem at Instituto Nacional de Emergência Médica (INEM) in Portugal. Established in 1981 under the Ministry of Health, this public institution aims to guarantee highly specialised health care services in emergency situations (Ministério da Saúde, 1981). Their actions include reception of the emergency request, prompt and accurate assistance at the scene when necessary, and aided transportation to the convenient health facility. In order to deliver pre-hospital care at the scene and during transportation in life-threatening situations, INEM requires highly qualified professionals and specialised material resources (INEM, 2017). INEM comprises a dispatch cen-

* Corresponding author at: Faculty of Economics and Business, Department of Information Management, Modeling and Simulation, KU Leuven Campus Brussels, Warmoesberg, 26, Brussels, 1000 Belgium.

E-mail addresses: hendrik.vermuyten@kuleuven.be (H. Vermuyten), joana.namorado.rosa@gmail.com (J. Namorado Rosa), ines.marques.p@tecnico.ulisboa.pt (I. Marques), jeroen.belien@kuleuven.be (J. Beliën), apovoa@tecnico.ulisboa.pt (A. Barbosa-Póvoa).

https://doi.org/10.1016/j.eswa.2018.06.017
0957-4174/© 2018 Elsevier Ltd. All rights reserved.

ter and emergency vehicles, which are based in different locations. Both are staffed by a shared pool of technical personnel who each have a different set of skills to perform certain tasks. Furthermore, workers belong to different teams which each have their specific set of tasks and geographical location. However, workers can perform tasks in other teams to help meet staffing requirements. The problem then consists of finding a schedule that meets the staffing requirements on each shift, while ensuring a high level of employee satisfaction.

Despite the high complexity in building rosters, several institutions still plan their timetables manually (e.g. the real case-study addressed later on at INEM). This requires a lot of work from the administrative personnel. Moreover, it is not easy for human planners to take all the different constraints and objectives into account. By contrast, we will show how an expert system can provide solutions in significantly less time, while simultaneously improving the solution quality. Additionally, it improves transparency as the rules the algorithm uses to build a schedule are agreed upon beforehand, increasing employee perception about fairness of the resulting rosters. Finally, the decision support system can help managers analyse the impact of different factors, such as the number of available personnel or the skills level of the different workers, on different key performance indicators (KPIs) of the resulting timetables. This can help the organisation make decisions on e.g. hiring additional personnel or providing extra training to workers.

The contribution of this paper is three-fold. First, we introduce a new health care staff scheduling problem that arises at an EMS system which involves scheduling both technical and medical personnel. While scheduling medical staff (like nurses or physicians) has been widely studied, the literature on scheduling technical personnel for medical services is much scarcer. The problem under consideration is unique in the sense that a geographically dispersed workforce needs to be assigned to different services, teams and tasks taking into account skills and workforce regulations. We show how one integrated model can optimise the schedule for this (partly) shared personnel at different locations. Second, an extensive computational study demonstrates that a VNDS heuristic outperforms a standard IP model and diving heuristic for this type of problem. Third, the successful application of the VNDS heuristic on real-life data contributes to decreasing the gap that exists today between theory and practice in automated timetabling and scheduling.

The remainder of this paper is structured as follows. Section 2 presents an overview of the literature on staff scheduling problems, with an emphasis on EMS systems in particular. Next, the staff scheduling problem at INEM is explained in Section 3, followed by two different solution approaches in Section 4. In Section 5, the model is applied to a real-life dataset provided by INEM as well as a test set of problems of varying dimensions. Finally, Section 6 concludes the paper.

2. Literature review

Personnel scheduling problems arise in a wide variety of settings, such as transportation systems, call centers, and health care systems (Ernst et al., 2004b). Different reviews of the literature have been published. Brucker, Qu, and Burke (2011) present generic mathematical programming formulations for permanence and fluctuation centred planning and for two special cases. They focus on complexity and discuss some polynomially solvable cases. Van Den Bergh, Beliën, De Bruecker, Demeulemeester, and De Boeck (2013) look at four different sets of problem characteristics, namely (i) personnel characteristics, decision delineation, and shifts definition, (ii) constraints, performance measures, and flexibility, (iii) solution method and uncertainty incorporation, and (iv) application and applicability of research. Finally, an annotated bibliography of ca. 700 articles with a short summary of each paper is presented by Ernst, Jiang, Krishnamoorthy, Owens, and Sier (2004a). The papers are classified according to the type of problem addressed, the application areas covered, and the methods used.

Health care systems have played a dominant role in the automated personnel scheduling research literature, mainly due to the nurse scheduling problem (NSP) which has been studied extensively. Reviews of models and solution methods for the NSP are provided by Cheang, Li, Lim, and Rodrigues (2003) and Burke, Causmaecker, Vanden Berghe, and Landeghem (2004). Besides nurses, the scheduling of other types of medical personnel like physicians have been subject to many research studies (Erhard, Schoenfelder, Fügener, & Brunner, 2017). Although there is a broad literature on EMS systems, most of the research efforts have focused on EMS location problems. Li, Zhao, Zhu, and Wyatt (2011) review the different EMS covering models and solution techniques, while Başar, Çatay, and Ünlüyurt (2012) present a taxonomy of EMS location problems classifying the existing literature with respect to problem type, modeling, and methodology. Ingolfsson (2013) presents a broader review on managing EMS systems, including other aspects like measuring performance and forecasting demand, workload and response times. Henderson (2011) outlines some of the key challenges in managing EMS systems and discusses how system-status management can serve as a tool to address several of these challenges.

Despite the large number of research papers on workforce scheduling on the one hand and EMS systems on the other hand, only a few papers deal with workforce management related to EMS systems (Aringhieri, Bruni, Khodaparasti, & van Essen, 2017). Addis, Aringhieri, Carello, Grosso, and Maffioli (2012) study the EMS system of Milano in which operators need to be assigned to teams and teams to predefined shifts. Whereas we also take into account workers at EMS vehicles in neighboring regions, Addis et al. (2012) focus on the operators working at the EMS system operation center only. As a result, the considered problem can be solved using a state-of-the-art commercial solver avoiding the need to develop heuristic algorithms. Similar to our study, Bradbeer, Findlay, and Fogarty (2000) assume that the number and locations of the EMS vehicles are fixed. They propose an evolutionary algorithm to develop personnel rosters for the EMS vehicles. In contrast to the problem addressed in this paper, other studies have dealt with the integrated EMS location problem and workforce scheduling problem. Erdoan, Erkut, Ingolfsson, and Laporte (2010) propose a solution method for the combined problem of locating ambulances and scheduling the working hours of ambulance crews in order to maximise the expected coverage with probabilistic response times. Similarly, Rajagopalan, Saydam, Sharer, and Setzler (2011) determine in a first stage the minimal number and location of ambulances per time interval such that expected coverage constraints are met. In the second stage actual crew schedules are developed. Li and Kozan (2009) also apply a two-stage approach. In the first stage, the shift start times and the number of ambulance staff required in each shift are determined, while the second stage entails assigning individuals to the first-stage generated shifts in order to achieve a monthly (four weeks) schedule at minimal cost. Finally, Vile, Gillard, Harper, and Knight (2016) develop several techniques, ranging from demand forecasting to optimisation methods for determining minimum staffing requirements and generate low-cost rosters, that are integrated in a master workforce capacity planning tool for the EMS system in Wales.

Many solution techniques are applied to solve personnel scheduling problems. A straightforward approach is to model the problem as an integer programming formulation and solve it using a general IP solver (Bard, Binici, & DeSilva, 2003;

Isken, 2004). Other researchers have used branch-and-price (Bard & Purnomo, 2005a; 2005b; Burke & Curtois, 2010), which uses column generation in each node in the branch-and-bound tree to solve the LP relaxation (Hans, 2001).

Despite the vast improvements in computer hardware and commercial IP solvers in the last decades, staff scheduling problems remain difficult to solve to optimality. Furthermore, optimal solutions that require many hours to calculate are often less valuable than quick suboptimal solutions, which allow user feedback or sensitivity analysis (Cheang et al., 2003). Also, heuristics are relatively easy to implement and are able to deal with complex constraints or objectives (e.g. non-linear cost functions) (Ernst et al., 2004b). This has led researchers to the application of heuristics to personnel scheduling problems. Genetic algorithms are a popular type of metaheuristic in this field (Aickelin & Dowsland, 2000; 2004; Bradbeer et al., 2000; Moz & Vaz Pato, 2007; Pato & Moz, 2008; Puente, Gómez, Fernández, & Priore, 2009). Other approaches include tabu search (Bellanti, Carello, Della Croce, & Tadei, 2004), iterated local search (Bellanti et al., 2004), particle swarm optimisation (Altamirano, Riff, Araya, & Trilling, 2012), memetic algorithms combining genetic algorithms with local improvement procedures (Burke, Cowling, De Causmaecker, & Vanden Berghe, 2001), and hyperheuristics (Smet, Bilgin, De Causmaecker, & Vanden Berghe, 2014). Variable neighbourhood search (VNS) is an increasingly popular metaheuristic for solving difficult personnel scheduling problems (Rahimian, Akartunal, & Levine, 2017; Zheng, Liu, & Gong, 2017). VNS is founded on the idea of using different neighbourhood structures to avoid getting stuck in local optima, since a local optimum in one neighbourhood is not necessarily a local optimum in another neighbourhood.

Recently, MIP-based heuristics have been successfully applied to the NSP (Burke, Li, & Qu, 2010; Santos, Toffolo, Gomes, & Ribas, 2016; Valouxis, Gogos, Goulas, Alefragis, & Housos, 2012). These approaches aim to combine the strengths of both mixed integer programming and (meta)heuristic approaches. Burke et al. (2010) first solve the NSP using an IP model that includes all hard constraints and only a subset of soft constraints. Then in a second step, the solution obtained by the IP model is improved using a variable neighbourhood search which focuses on the satisfaction of the other soft constraints. Valouxis et al. (2012) also use a two-phase approach. In the first phase, they assign nurses to days without considering shifts. The planning period is divided into groups of seven days, and for each group an IP model is solved. Next, the resulting solution is improved using local search by recombining partial schedules. In the second phase, for each day an IP model is solved that assigns nurses to shifts. This phase only takes costs related to shift assignments into account. This sequence is repeated until the available computation time is exhausted. Santos et al. (2016) use a two-pronged method, consisting of cut generation to improve bounds in order to prove optimality and the use of primal heuristics to quickly find good solutions. Starting from a feasible initial solution, they use a variable neighbourhood descent consisting of two different neighbourhood moves to improve the solution. The first neighbourhood fixes all assignments except for a number of days. Then this subproblem is solved to optimality using integer programming. Next, the sequence is repeated for a different subset of days. In a second step, the second neighbourhood fixes all allocations of shifts, except one. Again, for every shift a subproblem is solved using integer programming.

Instead of using standard MIP approaches inside a heuristic framework, column generation can be used as well (Gamache, Soumis, Marquis, & Desrosiers, 1999; Gomes, Toffolo, & Santos, 2017; Joncour, Michel, Sadykov, Sverdlov, & Vanderbeck, 2010). Gomes et al. (2017) combine column generation with variable neighbourhood search to efficiently find columns with negative reduced costs. Diving heuristics are used to obtain integer feasible solutions. Diving heuristics heuristically select branches in the branch-and-price tree using a certain rounding strategy such as rounding down, up, to the closest integer, or based on a threshold, until the first integer solution is found (Joncour et al., 2010).

3. Problem statement

The problem formulation in this paper is based on the real-life staff scheduling problem encountered at INEM in Portugal. INEM consists of two main services: the emergency dispatch centres known as Centro de Orientação de Doentes Urgentes (CODUs) and the Emergency Vehicles (EVs). CODUs are responsible for establishing a link between the caller requesting emergency medical assistance and the EVs by acquiring the calls, delivering voice assistance and instructions, and dispatching the convenient transportation. Both services operate 24/7.

The CODUs and the EVs share the same workforce of Técnicos de Emergência Pré-Hospitalar (TEPHs), i.e. technical personnel. INEM only schedules the set of TEPHs which is, in fact, the largest group of emergency staff. Medical doctors and psychologists are fewer in number and their schedule is built by the institution they belong to, i.e. hospitals or other health care units. The nurses are scheduled by INEM, but this schedule is much simpler than that of the TEPHs due to the significantly smaller size of this group.

Each TEPH is allocated to one of the two services, the CODUs or the EVs. Within their service, TEPHs belong to teams. TEPHs can belong to only a single team or to multiple teams. Each team has a set of tasks that need to be performed. The assignment of TEPHs to a team depends on two main factors. First, a TEPH can only perform tasks if (s)he has the required skills, e.g. a licence to drive an emergency motorcycle. Second, TEPHs' residence place must be taken into account in the attribution of tasks, e.g. people cannot be assigned to teams from locations that are more than 100 km away. TEPHs are expected to perform tasks in their respective team(s), but they can be assigned to tasks from other teams or services to meet task demands if needed. However, this should be avoided if possible.

The primary objective of the schedule is to ensure functionality of the services. Each day *d* of the planning horizon is divided into three shifts in which tasks are performed: a night shift from midnight to 8 am, a morning shift from 8 am to 4 pm, and an afternoon shift from 4 pm to midnight. However, the model allows that some tasks have a duration that differs from the shift length. The task then starts at the same time as the shift it is assigned to, but can finish either before or after the end of the shift. Workforce demands are determined in advance and can vary between shifts. Legal rules require a minimum resting time of two shifts between each pair of shifts worked. In addition, working time regulations require a minimum number of Sundays off over the planning period. They also put a limit on the maximum number of consecutive days worked and the maximum number of consecutive days off.

Secondly, the model intends to be equitable for every person. First, every person needs to work at least a predetermined number of night shifts, morning shifts, and afternoon shifts. Second, people prefer to have the entire weekend off instead of a single day. Third, each worker's contract hours should be met as much as possible, meaning both overtime and 'undertime' are undesirable. Finally, the number of tasks assigned to people from other teams should be minimised. These last three constraints are soft constraints, while the others are hard constraints.

The following integer programming formulation defines our problem. We use the following notation:

- Sets
  - $p \in P$: the set of people

- $t \in \mathrm{T}$: the set of tasks
- $d \in \mathrm{D}$: the set of days in the planning horizon
- $w \in \mathrm{W}$: the set of full weekends in the planning horizon
- $s \in \mathrm{S}$: the set of shifts, i.e. $\mathrm{S} = \{\text{night, morning, afternoon}\}$
- $g \in \mathrm{G}$: the set of teams
- $j \in \mathrm{J}$: the set of services

- Subsets
  - $\mathrm{P}^{\mathrm{T}}_t$: people that can perform task t
  - $\mathrm{P}^{\mathrm{G}}_g$: people that belong to team g
  - $\mathrm{T}^{\mathrm{P}}_p$: tasks that can be performed by person p
  - $\mathrm{T}^{\mathrm{G}}_g$: tasks that belong to team g
  - $\mathrm{T}^{\mathrm{J}}_j$: tasks that belong to service j
  - G_j: teams that belong to service j
- Parameters
  - κ: start day of the planning horizon (0 = Monday, 1 = Tuesday, ... , 6 = Sunday)
  - Θ_p: the number of contract hours of person p, adjusted for holidays
  - L_t: the duration of task t
  - R_{tds}: the required number of people assigned to task t on shift s of day d
  - θ^1: maximum number of consecutive working days
  - θ^2: maximum number of consecutive days off
  - θ^3: minimum number of Sundays off
  - θ^4_s: minimum number of shifts of type s worked
  - w^{RE+}_j: weight of penalty variable for excess workforce supply in service j
  - w^{RE-}_j: weight of penalty variable for shortage workforce supply in service j
  - w^{WO}: weight of penalty variable for full weekend off
  - w^{H+}: weight of penalty variable for excess hours worked
  - w^{H-}: weight of penalty variable for shortage hours worked
  - w^{G}_j: weight of penalty variable for assigning tasks of a team to members of another team in service j
- Decision variables
  - $x_{ptds} \in \{0,1\}$: equals 1 if person p is assigned to task t on shift s of day d, 0 otherwise
  - $Y^{RE+}_{tds} \in \mathbb{N}$: penalty variable for excess workforce supply for task t on shift s of day d
  - $Y^{RE-}_{tds} \in \mathbb{N}$: penalty variable for shortage workforce supply for task t on shift s of day d
  - Y^{WO+}_{pw}, $Y^{WO-}_{pw} \in \mathbb{N}$: penalty variables for full weekend off for person p in weekend w
  - $Y^{H+}_p \in \mathbb{N}$: penalty variable for excess hours worked for person p
  - $Y^{H-}_p \in \mathbb{N}$: penalty variable for shortage hours worked for person p
  - $Y^{G}_g \in \mathbb{N}$: penalty variable for assigning tasks of team g to members of another team

Now, the IP formulation can be written as follows:

$$\text{minimise} \quad \sum_{d\in \mathrm{D}}\sum_{s\in \mathrm{S}}\sum_{j\in \mathrm{J}}\sum_{t\in \mathrm{T}^{\mathrm{J}}_j}\left(\mathrm{w}^{\mathrm{RE+}}_{\mathrm{j}}\,\mathrm{Y}^{\mathrm{RE+}}_{\mathrm{tds}} + \mathrm{w}^{\mathrm{RE-}}_{\mathrm{j}}\,\mathrm{Y}^{\mathrm{RE-}}_{\mathrm{tds}}\right) + \sum_{p\in \mathrm{P}}\sum_{w\in \mathrm{W}}\left(w^{WO}\,Y^{WO+}_{pw} + w^{WO}\,Y^{WO-}_{pw}\right) + \sum_{p\in \mathrm{P}}\left(w^{H+}\,Y^{H+}_p + w^{H-}\,Y^{H-}_p\right) + \sum_{j\in \mathrm{J}}\sum_{g\in \mathrm{G}_j} w^{G}_j\,Y^{G}_g \tag{1}$$

$$\sum_{p\in \mathrm{P}^{\mathrm{T}}_t} x_{ptds} - Y^{RE+}_{tds} + Y^{RE-}_{tds} = R_{tds} \quad \forall t \in \mathrm{T}, \mathrm{d} \in \mathrm{D}, \mathrm{s} \in \mathrm{S} \tag{2}$$

$$\sum_{t\in \mathrm{T}^{\mathrm{P}}_p}\left(x_{ptd,\text{night}} + x_{ptd,\text{morning}} + x_{ptd,\text{afternoon}}\right) \le 1 \quad \forall p \in \mathrm{P}, \mathrm{d} \in \mathrm{D} \tag{3}$$

$$\sum_{t\in \mathrm{T}^{\mathrm{P}}_p}\left(x_{ptd,\text{morning}} + x_{ptd,\text{afternoon}} + x_{pt,d+1,\text{night}}\right) \le 1 \quad \forall p \in \mathrm{P}, \mathrm{d} \in \mathrm{D} \setminus \{|\mathrm{D}|\} \tag{4}$$

$$\sum_{t\in \mathrm{T}^{\mathrm{P}}_p}\left(x_{ptd,\text{afternoon}} + x_{pt,d+1,\text{night}} + x_{pt,d+1,\text{morning}}\right) \le 1 \quad \forall p \in \mathrm{P}, \mathrm{d} \in \mathrm{D} \setminus \{|\mathrm{D}|\} \tag{5}$$

$$\sum_{t\in \mathrm{T}^{\mathrm{P}}_p}\sum_{r\in\{d,d+1,\dots,d+\theta^1\}}\sum_{s\in \mathrm{S}} x_{ptrs} \le \theta^1 \quad \forall p \in \mathrm{P}, \mathrm{d} \in \mathrm{D} \setminus \{|\mathrm{D}|, |\mathrm{D}|-1, \dots, |\mathrm{D}| - \theta^1 + 1\} \tag{6}$$

$$\sum_{t\in \mathrm{T}^{\mathrm{P}}_p}\sum_{r\in\{d,d+1,\dots,d+\theta^2\}}\sum_{s\in \mathrm{S}} x_{ptrs} \ge 1 \quad \forall p \in \mathrm{P}, \mathrm{d} \in \mathrm{D} \setminus \{|\mathrm{D}|, |\mathrm{D}|-1, \dots, |\mathrm{D}| - \theta^2 + 1\} \tag{7}$$

$$\sum_{t\in \mathrm{T}^{\mathrm{P}}_p}\sum_{d\in\{7-\kappa,7-\kappa+7,\dots\}}\sum_{s\in \mathrm{S}} x_{ptds} \le |W| - \theta^3 \quad \forall p \in \mathrm{P} \tag{8}$$

$$\sum_{t\in \mathrm{T}^{\mathrm{P}}_p}\sum_{d\in \mathrm{D}} x_{ptds} \ge \theta^4_s \quad \forall p \in \mathrm{P}, \forall \mathrm{s} \in \mathrm{S} \tag{9}$$

$$\sum_{t\in \mathrm{T}^{\mathrm{P}}_p}\sum_{s\in \mathrm{S}} x_{pt,7(w-1)+7-\kappa,s} - x_{pt,7(w-1)+6-\kappa,s} - Y^{WO+}_{pw} + Y^{WO-}_{pw} = 0 \quad \forall p \in \mathrm{P}, \mathrm{w} \in \mathrm{W} \tag{10}$$

$$\sum_{t\in \mathrm{T}^{\mathrm{P}}_p}\sum_{d\in \mathrm{D}}\sum_{s\in \mathrm{S}} L_t x_{ptds} - Y^{H+}_p + Y^{H-}_p = \Theta_p \quad \forall p \in \mathrm{P} \tag{11}$$

$$\sum_{p\in \mathrm{P}^{\mathrm{G}}_g}\sum_{t\in \mathrm{T}^{\mathrm{P}}_p \setminus \mathrm{T}^{\mathrm{G}}_g}\sum_{d\in \mathrm{D}}\sum_{s\in \mathrm{S}} x_{ptds} - Y^{G}_g = 0 \quad \forall g \in \mathrm{G} \tag{12}$$

$$x_{ptds} \in \{0,1\} \quad \forall p \in \mathrm{P}, \mathrm{t} \in \mathrm{T}, \mathrm{d} \in \mathrm{D}, \mathrm{s} \in \mathrm{S} \tag{13}$$

$$Y^{RE+}_{tds}, Y^{RE-}_{tds} \in \mathbb{N} \quad \forall t \in \mathrm{T}, \mathrm{d} \in \mathrm{D}, \mathrm{s} \in \mathrm{S} \tag{14}$$

$$Y^{WO+}_{pw}, Y^{WO-}_{pw} \in \mathbb{N} \quad \forall p \in \mathrm{P}, \mathrm{w} \in \mathrm{W} \tag{15}$$

$$Y^{H+}_p, Y^{H-}_p \in \mathbb{N} \quad \forall p \in \mathrm{P} \tag{16}$$

$$Y^{G}_g \in \mathbb{N} \quad \forall g \in \mathrm{G} \tag{17}$$

The objective function (1) minimises the weighted sum of the penalty variables. Constraints (2) are the coverage requirements. Understaffing and overstaffing are allowed at a penalty cost. People need to rest 2 shifts between consecutive shifts worked, which is enforced by constraints (3)–(5) for night shifts, morning shifts, and afternoon shifts respectively. People are not allowed to work more than θ^1 days consecutively (6) and they cannot have more than θ^2 consecutive days off (7). Over the entire planning horizon, people should have at least θ^3 Sundays off (8). Every person needs to work at least θ^4_s shifts of each type s (9). Preferably, people get the entire weekend off instead of a single day (10). Ideally, people work their specified number of contract hours, adjusted for the number of holidays in the planning period (11). Finally, tasks belonging to a certain team should be assigned to members of that team as much as possible (12). This constraint works as follows: for every team g, the penalty variable Y^G_g counts the number of tasks from other teams that are assigned to members of this team. Finally, constraints (13)–(17) define the domains of the binary and integer decision variables.

4. Solution approach

As is shown in Section 5.4, solving IP formulation (1)–(12) with a state-of-the-art commercial IP solver turns out to be intractable for real-life instances. Therefore, heuristic approaches are needed to provide good solutions in reasonable computation times. Two different approaches are compared. Section 4.1 describes a diving heuristic based on column generation. Next, in Section 4.2 we develop VNDS heuristic that utilises the principles of fix-and-optimise.

4.1. Method 1: Diving heuristic

4.1.1. Column generation

The model consisting of Eqs. (1)–(12) can be formulated in a different way. For every person a work pattern can be defined as the tasks assigned to that person over the planning horizon. Such a work pattern is referred to as a *column*. Let a_{pktds} equal 1 if column k for person p assigns task t on shift s of day d and 0 otherwise. Also, let c_{pk} be the cost of column k for person p, based on the soft constraint violations for person p. These consist of the penalties for receiving only single weekend days off instead of entire weekends, deviations in working time from contract hours (both overtime and ‘undertime’), and the number of tasks assigned to this person from other teams than his or her own team. Finally, let K_p be the set of columns for person p. Then we can define the following decision variable: $z_{pk} \in \{0, 1\}$: equals 1 if column k is chosen for person p, 0 otherwise.

Our problem can then be formulated as follows:

$$\text{minimise} \quad \sum_{p\in \mathrm{P}}\sum_{k\in \mathrm{K}_p}\left(c_{pk}\, z_{pk}\right)+\sum_{d\in \mathrm{D}}\sum_{s\in \mathrm{S}}\sum_{j\in \mathrm{J}}\sum_{t\in \mathrm{T}^{\mathrm{J}}_j}\left(w^{RE+}_j\, Y^{RE+}_{tds}+w^{RE-}_j\, Y^{RE-}_{tds}\right) \tag{18}$$

$$\sum_{p\in \mathrm{P}^{\mathrm{T}}_t}\sum_{k\in \mathrm{K}_p} a_{pktds}\, z_{pk}-Y^{RE+}_{tds}+Y^{RE-}_{tds}=R_{tds} \quad \forall t\in \mathrm{T}, d\in \mathrm{D}, s\in \mathrm{S} \tag{19}$$

$$\sum_{k\in \mathrm{K}_p} z_{pk}=1 \quad \forall p\in \mathrm{P} \tag{20}$$

$$z_{pk}\in\{0,1\} \quad \forall p\in \mathrm{P}, k\in \mathrm{K}_p \tag{21}$$

$$Y^{RE+}_{tds}, Y^{RE-}_{tds}\in \mathbb{N} \quad \forall t\in \mathrm{T}, d\in \mathrm{D}, s\in \mathrm{S} \tag{22}$$

The objective function (18) minimises the costs of the chosen work patterns and the understaffing and overstaffing costs. Constraints (19) are the coverage requirements. Constraints (20) enforce that exactly one work pattern is chosen for each person. Finally, constraints (21) and (22) define the domains of the binary and integer decision variables.

Column generation was first developed by Dantzig and Wolfe (1960) as a method for solving linear programming problems with an exponential number of variables. In the column generation method, the LP relaxation of formulations (18)–(22) is initialised with only a limited set of columns. Subsequently, at each iteration for every person a subproblem is solved to generate new columns until no more columns with negative reduced costs can be found (Hans, 2001). To generate new columns, a pricing problem is solved which takes into account the constraints related to an individual.

Let λ_{tds} represent the dual costs associated with constraints (19) and μ_p the dual costs associated with constraints (20). The reduced cost (RC) of a new column k for person p is then given by:

$$c_{pk}-\textstyle\sum_{t\in \mathrm{T}}\sum_{d\in \mathrm{D}}\sum_{s\in \mathrm{S}} a_{pktds}\, \lambda_{tds}-\mu_p. \tag{23}$$

We define the following decision variable for the pricing problem: $a_{tds} \in \{0, 1\}$ equals 1 if task t is assigned on shift s of day d, 0 otherwise.

The pricing problem for each person p can then be defined as follows:

$$\text{minimise} \quad \sum_{t\in \mathrm{T}}\sum_{d\in \mathrm{D}}\sum_{s\in \mathrm{S}}\left(-\lambda_{tds}\, a_{tds}\right) + \sum_{w\in \mathrm{W}}\left(w^{WO}\, Y^{WO+}_w+w^{WO}\, Y^{WO-}_w\right) + w^{H+}\, Y^{H+}+w^{H-}\, Y^{H-}+\sum_{j\in \mathrm{J}}\sum_{g\in \mathrm{G}_j} w^G_j\, Y^G_g \tag{24}$$

$$\sum_{t\in \mathrm{T}^{\mathrm{P}}_p}\left(a_{td,\mathrm{night}}+a_{td,\mathrm{morning}}+a_{td,\mathrm{afternoon}}\right)\leq 1 \quad \forall d\in \mathrm{D} \tag{25}$$

$$\sum_{t\in \mathrm{T}^{\mathrm{P}}_p}\left(a_{td,\mathrm{morning}}+a_{td,\mathrm{afternoon}}+a_{t,d+1,\mathrm{night}}\right)\leq 1 \quad \forall d\in \mathrm{D}\setminus\{|\mathrm{D}|\} \tag{26}$$

$$\sum_{t\in \mathrm{T}^{\mathrm{P}}_p}\left(a_{td,\mathrm{afternoon}}+a_{t,d+1,\mathrm{night}}+a_{t,d+1,\mathrm{morning}}\right)\leq 1 \quad \forall d\in \mathrm{D}\setminus\{|\mathrm{D}|\} \tag{27}$$

$$\sum_{t\in \mathrm{T}^{\mathrm{P}}_p}\sum_{r\in\{d,d+1,\dots,d+\theta^1\}}\sum_{s\in \mathrm{S}} a_{trs}\leq \theta^1 \quad \forall d\in \mathrm{D}\setminus\{|\mathrm{D}|, |\mathrm{D}|-1,\dots,|\mathrm{D}|-\theta^1+1\} \tag{28}$$

$$\sum_{t\in \mathrm{T}^{\mathrm{P}}_p}\sum_{r\in\{d,d+1,\dots,d+\theta^2\}}\sum_{s\in \mathrm{S}} a_{trs}\geq 1 \quad \forall d\in \mathrm{D}\setminus\{|\mathrm{D}|, |\mathrm{D}|-1,\dots,|\mathrm{D}|-\theta^2+1\} \tag{29}$$

$$\sum_{t\in \mathrm{T}^{\mathrm{P}}_p}\sum_{d\in\{7-\kappa,7-\kappa+7,\dots\}}\sum_{s\in \mathrm{S}} a_{tds}\leq |W|-\theta^3 \tag{30}$$

$$\sum_{t\in \mathrm{T}^{\mathrm{P}}_p}\sum_{d\in \mathrm{D}} a_{tds}\geq \theta^4_s \quad \forall s\in \mathrm{S} \tag{31}$$

$$\sum_{t\in \mathrm{T}^{\mathrm{P}}_p}\sum_{s\in \mathrm{S}} a_{t,7(w-1)+7-\kappa,s}-a_{t,7(w-1)+6-\kappa,s}-Y^{WO+}_w+Y^{WO-}_w = 0 \quad \forall w\in \mathrm{W} \tag{32}$$

$$\sum_{t\in \mathrm{T}^{\mathrm{P}}_p}\sum_{d\in \mathrm{D}}\sum_{s\in \mathrm{S}} L_t a_{tds}-Y^{H+}+Y^{H-}=\Theta_p-\eta\zeta \tag{33}$$

$$\sum_{t\in \mathrm{T}^{\mathrm{P}}_p\setminus \mathrm{T}^{\mathrm{G}}_g}\sum_{d\in \mathrm{D}}\sum_{s\in \mathrm{S}} a_{tds}-Y^G_g=0 \quad \forall g\in \mathrm{G}, \text{if } p\in \mathrm{G} \tag{34}$$

$$Y^{WO+}_w, Y^{WO-}_w\in \mathbb{N} \quad \forall w\in \mathrm{W} \tag{35}$$

$$Y^{H+}, Y^{H-}\in \mathbb{N} \tag{36}$$

$$Y^G_g\in \mathbb{N} \quad \forall g\in \mathrm{G} \tag{37}$$

4.1.2. Column addition

The column generation loop can be implemented in different ways (see e.g., Beliën & Demeulemeester, 2006). We consider two possibilities. A first possibility is to solve the master problem, obtain the dual variables, and then solve one subproblem for every person p. All columns that are found with a negative reduced cost are added to the master problem, after which a new iteration begins. A second possibility is to solve the master problem and then solve a subproblem for person p. If a column with negative reduced cost is found, it is added to the master and the master is reoptimised. Then we solve the subproblem for person $p+1$ and so on.

Algorithm 1. Diving heuristic.

```
Require: branching threshold $\delta$
1: initiate master with $|P|$ supercolumns
2: integer_solution_found ← false
3: while integer_solution_found = false do
4:     obj ← column_generation()
5:     if all $z_{pk}$ integer then
6:         integer_solution_found ← true
7:     else
8:         for all $z_{pk} > \delta$ do
9:             set $z_{pk}$ to 1
10:            remove all $z_{pk'}$ with $k' \neq k$ from master problem
11:        end for
12:        if no $z_{pk} > \delta$ exist then
13:            set largest $z_{pk}$ to 1
14:            remove all $z_{pk'}$ with $k' \neq k$ from master problem
15:        end if
16:    end if
17: end while
```

4.1.3. Finding an integer solution

The solution found using column generation might not be integral. Therefore, branching is needed to find an integer solution. However, as an exact branch-and-price approach requires a prohibitively large amount of computation time for large problem instances, a diving heuristic is used to quickly find good solutions. In a diving heuristic, the branch-and-price tree is traversed in a depth-first manner. After solving the LP relaxation, one or more variables are selected for branching. In our implementation, all fractional variables with a value above a certain threshold δ are fixed to one. While branching on the column variables is usually not well-suited in a branch-and-price approach because it generates an imbalanced tree and requires significant changes to the pricing problems (Vanderbeck, 2000), in a diving heuristic the former is actually advantageous as this reduces the remaining solution space much more and the latter poses no problem as variables are always fixed to one and never to zero (Joncour et al., 2010). Like in a regular branch-and-price approach, after each branching decision, new columns are generated as it is not guaranteed that the columns of the root node can be combined into good integer solutions (Beliën & Demeulemeester, 2006). Subproblems naturally only need to be solved for those people p for which no column variable has been fixed yet. Also, all columns for people for which a column has been fixed are explicitly removed from the master problem (Gamache et al., 1999). Algorithm 1 presents the pseudo-code for the diving heuristic.

4.2. Method 2: VNDS heuristic

Fix-and-optimise heuristics (also referred to as MIP-heuristics) have been increasingly applied to solve a wide variety of scheduling problems (e.g., Dorneles, De Araújo, & Buriol, 2014; Helber & Sahling, 2010; Santos et al., 2016; Seeanner, Almada-Lobo, & Meyr, 2013). The idea is that a MIP solver iteratively optimises subproblems in which only a subset of variables is free (i.e., can be changed) while all other variables are fixed to their value in the current solution. Since the solver takes the entire optimisation problem into account, a new solution is always feasible and at least as good as the previous solution. An important consideration is which and how many variables are released in each subproblem. The larger a subproblem, the higher the probability that an improved solution is found, but this comes at an increased computational cost.

VNDS was developed by Hansen, Mladenović, and Perez-Britos (2001) and is a variant of VNS (Hansen & Mladenović, 2001). An important idea behind VNS is the use of different neighbourhoods to improve the solution. VNS consists of a local search (first descent) phase and a shake phase, which are repeated until the stopping criterion is met. Local search is used to intensify the search whereas the shake phase brings diversity. If the local search phase gets stuck in a local optimum, VNS moves to the shake phase, in which a random neighbour of the current solution is selected. The algorithm then reverts to the local search phase. As the search progresses, the shake phase uses different neighbourhoods to explore solutions further removed from the current solution. The change of neighbourhoods can also be performed during the local search phase, which is referred to as variable neighbourhood descent (VND). In contrast to VNS, VNDS divides a neighbourhood into different subproblems in which only a subset of variables is considered, instead of the entire neighbourhood simultaneously.

The neighbourhoods for the local search phase can be searched in different ways by the fix-and-optimise heuristic. Santos et al. (2016) use a fixed sequence of two neighbourhoods. The first neighbourhood is divided into a fixed number of subproblems, which are solved in sequence. They only explore the second neighbourhood if the first neighbourhood finds a better solution. The search then goes back to the first neighbourhood and increases the size of the subproblems. A second method is proposed by James and Almada-Lobo (2011), who also consider two neighbourhoods. The first neighbourhood is explored until a local optimum is reached. Then the second neighbourhood is searched and the algorithm reverts to the first one. Instead of solving a fixed sequence of subproblems in the first neighbourhood, they randomly select a subproblem with probabilities based on the frequency and recency with which the different subproblems have previously been selected. The size of the subproblems stays the same over the search. Camargo, Toledo, and Almada-Lobo (2014) choose a partition based on how often variables have changed during the search process. Variables which have changed often between different incumbent solutions are more likely to be chosen compared to variables which have retained mostly the same value, as the latter are more likely to already have the value they would have in the optimal solution. Finally, Belo-Filho, Amorim, and Almada-Lobo (2015) randomly select different neighbourhoods during the search based on how often the neighbourhood has succeeded in improving the solution. The probability of selecting a neighbourhood increases if it has been more successful. The sizes of subproblems selected from the neighbourhood are increased or decreased during the search based on their required computation time and the optimality gap of the current solution.

In the next section, we give an overview of the proposed VNDS implementation.

4.2.1. Overview

The VNDS is used to control the search, with a shake phase to escape local optima. The fix-and-optimise heuristic is used for the local search phase and incorporates the idea of using different neighbourhoods to iteratively improve the solution. Algorithm 2 presents the pseudo-code for the VNDS heuristic.

In the proposed implementation, each neighbourhood has a fixed probability of being chosen at each iteration. This setup has the advantage of randomness in the neighbourhood sequence, which helps to avoid getting stuck in local optima, while only a single set of parameters has to be specified as no updating scheme has to be provided.

Section 4.2.2 first describes the constructive method, followed by the neighbourhood structures used in the local search phase in Section 4.2.3 and finally the shake phase in Section 4.2.4.

Algorithm 2. VNDS heuristic.

```
Require: neighbourhoods $\mathcal{N}^k$ and probabilities $\pi^k$; max_iter
1: $S \leftarrow$ constructive_method()
2: $S' \leftarrow S$
3: while no timeout do
4:     iter $\leftarrow 0$
5:     while iter $<$ max_iter do
6:         choose neighbourhood $\mathcal{N}^k$ based on probabilities $\pi^k$
7:         choose a subproblem $\mathcal{P}$ in $\mathcal{N}^k$ and fix variables $x_{ptds}$ accordingly
8:         $S'' \leftarrow$ optimise_subproblem($\mathcal{P}(S')$)
9:         if obj($S''$) $<$ obj($S'$) then
10:            $S' \leftarrow S''$
11:            iter $\leftarrow 0$
12:        else
13:            iter $\leftarrow$ iter $+ 1$
14:        end if
15:    end while
16:    if obj($S'$) $<$ obj($S$) then
17:        $S \leftarrow S'$
18:    end if
19:    $S' \leftarrow$ shake($S$)
20: end while
```

4.2.2. Constructive method

The purpose of the constructive method is to find a feasible schedule to serve as a starting point for the VNDS. In the problem formulation of (1)–(12), all constraints are person-specific, except for the coverage requirements (2). Furthermore, since the coverage requirements allow both understaffing and overstaffing, these constraints do not impact the feasibility of a solution. As a result, a feasible solution can be obtained by solving a MIP model for every person, consisting of all constraints specific to that person, i.e. constraints (3)–(12). The objective function consists of the penalty values of the soft constraints (10)–(12) related to the given person. The solution of each individual MIP model gives a partial schedule with the task assignments for an individual person over the planning horizon. To ensure that as many tasks as possible are selected in each MIP and the solver does not generate nearly empty schedules, each task is given a large negative objective function coefficient (since it is a minimisation problem). If a partial schedule has been built for every person, a feasible start solution has been obtained. This constructive method can be used to build feasible start solutions for general staff scheduling problems, provided the only linking constraints are the demand coverage requirements and all other constraints are person-specific.

4.2.3. Local search phase

Three different neighbourhood structures are used in the local search phase:

Neighbourhood $\mathcal{N}_d^D$ This neighbourhood randomly selects d consecutive days. All assignments on these days are released, while other assignments are fixed. The idea behind this neighbourhood is that it can better match people with the required skills to tasks on the selected days because it considers all shifts and all people simultaneously. Increasing the number of free days d increases the probability of finding better solutions, but it also leads to increased computation times to solve the subproblem.

Neighbourhood $\mathcal{N}^S$ This second neighbourhood solves three subproblems consecutively. In the first subproblem, all assignments on the first shift of each day are freed. If the newly found solution is better, the search proceeds from this new solution. Next, the second subproblem frees all assignments on the second shift of each day. Finally, the third subproblem frees all assignments of the third shift of each day. While the size of a subproblem might seem quite large, it turns out that it can be solved relatively easily in practice. The reason is that the assignments of the other two shifts are fixed and people can only work one shift per day, so many possibilities can be immediately discarded.

Neighbourhood $\mathcal{N}_t^T$ This last neighbourhood randomly selects t different tasks. For each selected task, a subproblem is created in which all assignments are fixed, except for the assignments related to the selected task. These subproblems are then solved sequentially. Each time a solution is found that is better than the current one, the search proceeds from this new solution. Initial tests revealed that solving a separate subproblem for each selected task yielded better results than solving a single subproblem in which all selected tasks are freed simultaneously. This neighbourhood is useful to find small improvements and to diversify the search. The size parameter t again entails a trade-off between the probability of finding better solutions and the required computation time.

At each iteration, one of the neighbourhoods is chosen based on their predefined probabilities π_d^D, π^S, and π_t^T. To avoid that the same subproblem is solved again, the neighbourhoods $\mathcal{N}^S$ and $\mathcal{N}_t^T$ cannot be selected twice in a row. In case two neighbourhoods $\mathcal{N}_d^D$ are selected consecutively, the first and last day of the newly selected subproblem both have to be different from the first and last day of the previous subproblem. If the neighbourhood finds a better solution, the current solution is set to this improved solution. Then a new iteration begins. This phase continues until it does not succeed in improving the solution any further within a predetermined number of iterations, in which case the algorithm moves to the shake phase.

4.2.4. Shake phase

The shake phase in VNDS is used to escape local optima. It always starts from the best solution. The neighbourhood structure $\mathcal{N}_p^P$ used in our implementation randomly selects p people. Next, the solution is fixed, except for the assignments belonging to the selected people. Their partial schedules are changed randomly by solving the same person-specific MIP model of the constructive method of Section 4.2.2. This time the objective function weights for every task are chosen at random, so that each time a different schedule will be generated. The generated partial schedules are then integrated in the current solution. Because understaffing and overstaffing are allowed and all other person-specific constraints are satisfied by the person-specific MIP model, this phase always preserves the feasibility of the solution.

At the start of the algorithm p is set to one. If the local search phase succeeds in improving the solution returned by the shake phase, p is reset to one and the best solution is updated. Otherwise, p is increased by one, so that the probability of escaping the local optimum increases.

5. Computational results

The algorithms are coded in C++14 and compiled with Microsoft Visual Studio 2015. The callable library of ILOG CPLEX 12.6.2 is used as a MIP solver. All tests are executed on a PC with an Intel Core i5-5200U CPU of 2.20 GHz and 8 GB of RAM under the Windows 10 operating system. In all tests, the maximum computation time was set at 1 h, unless specified otherwise.

The C++ codes for the diving heuristic, the VNDS heuristic, and the instance generator of Section 5.2, as well as detailed information on all problem instances can be found at the following website: https://orcorner.wordpress.com/research-data/.

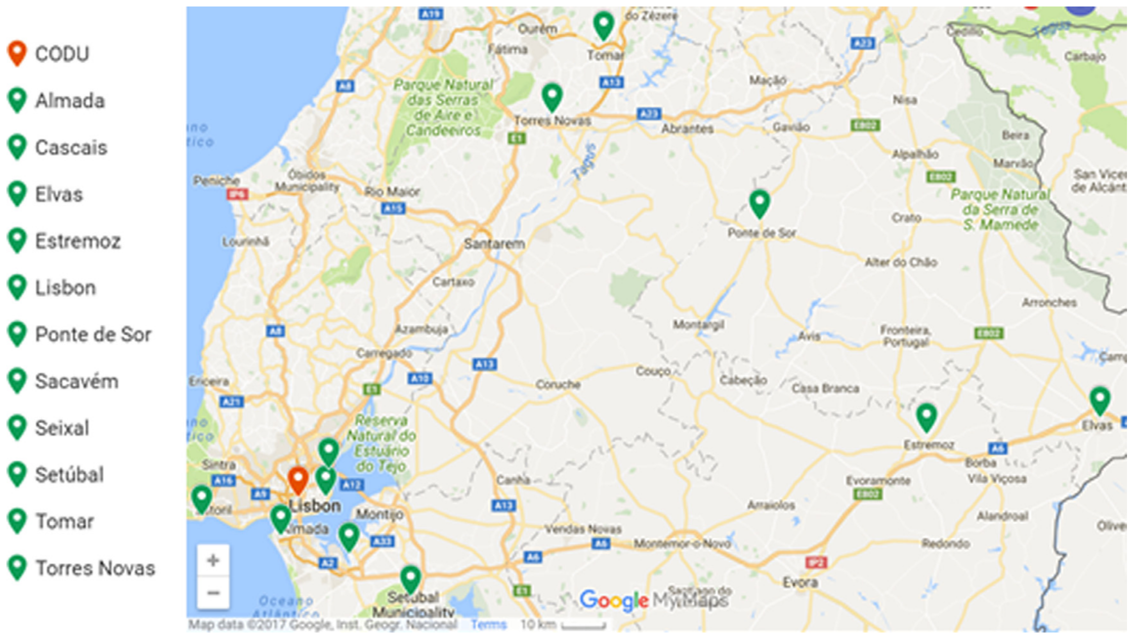


Fig. 1. Locations of the different INEM sites.

Table 1
Overview of the different types of tasks at INEM and their durations.

	CODU		EVs					
Task type	CODU Shift Responsible	CODU Task	AEM Driver	AEM Team Responsible	SIV Task	TIP Task	UMIPE Task	MEM Task
Duration				8 h				12 h

5.1. Case study at Instituto Nacional de Emergência Médica

While INEM has branches in every part of Portugal, the scope of our case study is the Lisbon region and some neighbouring regions, which are coordinated by the Lisbon CODU and where INEM has vehicles: Almada, Cascais, Elvas, Estremoz, Ponte de Sor, Sacavém, Setúbal, Seixal, Tomar, and Torres Novas (see Fig. 1).

The total workforce for all these branches consists of 289 TEPHs. They are divided into 22 teams, 5 in the CODU and 17 for the EVs in the different regions. In the CODU, two types of tasks exist: the regular CODU task which involves answering emergency calls and dispatching emergency medical transportation, and the shift responsibility task. The EVs consist of different types of ambulances, namely Medical Emergency Ambulances (AEM), Immediate Life-Support Ambulances (SIV), Inter-hospital Pediatric Transports (TIP), Mobile Units of Psychological Emergency Intervention (UMIPE), and Medical Emergency Motorcycles (MEM). Every type of ambulance requires one TEPH, except for the AEMs which require two TEPHs. This gives a total of eight types of tasks, which are summarised in Table 1. Each task has a duration equal to the shift length of 8 hours, except for the MEM tasks, which have a duration of 12 h. However, these tasks pose no problem in our model formulation, because they are only assigned to morning shifts and only impact the working time constraints (11). While the same types of tasks are done by multiple teams, they are represented by different tasks in the model. Since TEPHs should preferably be assigned to tasks within their own teams, each team has its own distinct set of tasks. As a result, there are 61 different tasks in the problem instance (10 for the CODU and 51 for the EVs). The planning horizon is four weeks (28 days). Over the entire planning horizon a total of 4527 task-demands (i.e., $\Sigma_{tds} R_{tds}$) need to be filled in. The problem dimensions are shown in Table 3.

Table 2
Objective function weights used in the computational tests.

Parameter	w^{RE+}	w^{RE-}_{CODU}	w^{RE-}_{EV}	w^{WO}	w^{H+}	w^{H-}	w^{G}_{CODU}	w^{G}_{EV}
Weight	10	100	1000	10	1	1	10	20

Additionally, it is necessary to account for the working time regulations. These state that people cannot work more than 6 days consecutively, cannot have more than 5 consecutive days off, and should have at least 1 Sunday off every four weeks. This implies that $\theta^1 = 6$ in constraints (6), $\theta^2 = 5$ in constraints (7), and $\theta^3 = 3$ in constraints (8). Furthermore, out of equity considerations with regards to shift distributions, it is required that each person works at least 2 shifts of each type, i.e. $\theta^4_s = 2, \forall s \in S$ in constraints (9). Finally, a standard contract specifies a working time of 140 hours per month, implying that $\Theta_p = 140, \forall p \in P$ in constraints (11).

The satisfaction of the task demands is considered the most important objective of the schedule. Therefore, w^{RE-}_{EV} and w^{RE-}_{CODU} are given the highest values. However, understaffing in the EVs is considered worse than understaffing in the CODU, as the CODU continues to operate without significant problems if there is a small shortage in personnel during a certain shift, thus $w^{RE-}_{EV} > w^{RE-}_{CODU}$. Secondly, because EVs are located in different regions, assigning people to tasks outside of their team (if allowed based on the distance of the location) is worse for the EVs than for the CODU, i.e. $w^{G}_{EV} > w^{G}_{CODU}$. The weights used in the computational tests are summarised in Table 2. Note that the dimensions of the various objectives are different, which needs to be taken into account when choosing these objective function weights. In Section 5.5, a sensitivity analysis is carried out to investigate the impact of changes in these weights.

Table 3
Summary of the instances. '# x var.' denotes the number of x_{ptds} variables. 'SL' is the skill level in percent. An SL of p percent means every person on average has the required skills for p percent of all tasks. Finally, 'util.' refers to the utilisation level in percent, i.e. the total demand per day per shift for all tasks divided by the available number of people, adjusted for available monthly working time.

Instance	|P|	|T|	|D|	# x var.	SL	util.
INEM	289	61	28	1,480,836	54	90
INEM MD	289	61	56	2,961,672	54	90
INEM MP	417	61	28	2,136,708	53	92
INEM LS	289	103	28	2,500,428	55	83
INEM HS	289	61	28	1,480,836	100	90
Test01	291	47	28	1,148,868	67	88
Test02	346	55	28	1,598,520	52	135
Test03	314	59	28	1,556,184	33	129
Test04	296	70	28	1,740,480	51	134
Test05	331	73	28	2,029,692	48	107
Test06	259	57	28	1,240,092	91	69
Test07	272	53	56	2,421,888	54	96
Test08	292	44	28	1,079,232	64	85
Test09	344	56	28	1,618,176	85	119
Test10	269	61	56	2,756,712	98	83
Test11	296	53	56	2,635,584	71	107
Test12	335	72	28	2,026,080	99	72
Test13	306	67	28	1,722,168	61	102
Test14	286	57	28	1,369,368	91	87
Test15	281	56	28	1,321,824	94	84

5.2. Test sets

In this section, the performance of the different algorithms is tested on problem instances with different dimensions. To the best of our knowledge, no instances exist in the literature that can be used to test our model without major changes. Therefore, a test set is generated and made publicly available.

First, four datasets are derived from the real data of INEM by changing one of the problem dimensions. These are respectively: INEM MD (more days), INEM MP (more people), INEM LS (less symmetry), and INEM HS (high symmetry). The INEM MD instance is obtained by extending the planning horizon from 28 to 56 days. The same task demands R_{tds} are used for both months. In the INEM MP instance the number of workers is increased to 417. This number is chosen so that each team's size increases by the same factor. The task demands R_{tds} are increased at the same rate as the number of workers. The INEM LS dataset is constructed by splitting tasks into two or more tasks, and assuming that people that have the skills to do the old task can only do one of the new tasks, so that symmetry is reduced. All task demands R_{tds} are divided proportionally over the new tasks. Finally, the INEM HS dataset is derived by assuming that every worker can do every task, so that symmetry is maximised. All other parameters are kept unchanged.

Next, an instance generator was developed that builds problem instances with random dimensions. The generator works as follows. First the problem dimensions are determined. The number of people |P|, the number of tasks |T|, and the number of teams |G| are drawn from uniform distributions $U(250, 350)$, $U(40, 80)$, and $U(5, 15)$ respectively. With a probability of 0.20 the number of days |D| is set to 56, otherwise |D| is set to 28. The maximal problem size is limited to 3,000,000 x_{ptds} variables, roughly double the size of the INEM instance. In a second step, each person p and task t are randomly assigned to a team g, as defined by the sets $\mathrm{P}_g^{\mathrm{G}}$ and $\mathrm{T}_g^{\mathrm{G}}$. The distribution of skills is determined by the overal skill level, which is the probability that a random person p has the required skills to do a certain task t, i.e. whether $p \in \mathrm{P}_t^{\mathrm{T}}$ and $t \in \mathrm{T}_p^{\mathrm{P}}$. This skill level is chosen from a uniform distribution $U(0.2, 1.0)$. In the third phase, demands R_{tds} are generated for each task t, day d, and shift s of the planning horizon. These demands are based on the number of people that have the required skills for a given task t (i.e., the supply), which is calculated as $|\mathrm{P}_t^{\mathrm{T}}|$, as well as the utilisation level. The utilisation level $\rho = \frac{\text{demand}}{\text{supply}}$ is chosen from a uniform distribution $U(0.6, 1.2)$. Since people can work only one shift per day and only 17.5 out of 28 days according to their contract hours, the supply per day per shift for task t (σ_t) is calculated as follows, $\sigma_t = \frac{17.5}{28}\,\frac{1}{3}\,|\mathrm{P}_t^{\mathrm{T}}|$. Finally, for each shift s of each day d, the actual demand R_{tds} for each task t is drawn from a Poisson distribution $P(\mu_t)$ with parameter $\mu_t = \rho\,\sigma_t$. Finally, all task durations L_t are set to 8 hours. The generator was used to produce an additional set of 15 instances, leading to a total of 20 instances to validate our algorithm. Information on these instances is summarised in Table 3.

5.3. Algorithm configurations and parameter settings

For the diving heuristic, initial tests revealed that the column generation phase suffers from the tailing-off effect, where a large number of iterations is required to find the optimal solution. To limit the total computation time to 1 h, the column generation phase is terminated before the LP optimum has been found. The allowed computation time for the root node is set at 1800 s, while the remaining 1800 s are used for all subsequent nodes combined. The reason for this choice is that increasing the computation time for the root node relative to that of other nodes allows the heuristic to find better solutions, since early branching decisions are never reversed and have a large impact on the search process later on.

For the VNDS heuristic, initial tests showed that for the neighbourhood $\mathcal{N}_t^T$, $t = 10$ provides a good trade-off between the size of the neighbourhood and the required computation time. If t is chosen smaller, $\mathcal{N}_t^T$ rarely succeeds in improving the solution, while for larger t, the required computation time becomes too large. Additionally, the maximum number of iterations without improvement in the local search phase (max_iter) is set to 10, since experiments showed that at that point the heuristic has usually converged to a local minimum. Moreover, the maximal computation time for a single subproblem is limited to 120 s.

In a second step, the performance of each of the individual neighbourhoods is investigated and compared with the performance of the heuristic when the different neighbourhoods are combined. All tests are executed on five representative problem instances with different sizes and characteristics, namely the INEM, INEM MP, Test03, Test05, and Test10 instances. For each instance, the VNDS is run three times, giving a total of fifteen observations for each of the configurations. The different configurations and their results on the five datasets are summarised in Table 4. The results show that the neighbourhoods $\mathcal{N}_d^D$ and $\mathcal{N}^S$ individually achieve the best results, while the $\mathcal{N}_t^T$ neighbourhood only finds small improvements, as was expected from the theoretical description in Section 4.2.3. Furthermore, while increasing d in neighbourhood $\mathcal{N}_d^D$ increases performance slightly for most instances, for the Test10 instance it actually deteriorates performance. The reason is that the computation time increases substantially when d is large and thus the heuristic fails to converge to a good solution in the given time limit. Therefore, the maximum size of neighbourhood $\mathcal{N}_d^D$ is limited to $d = 4$.

To examine the statistical differences in performance between the configurations, a non-parametric Kruskal–Wallis test is used, since the normality and homoscedasticity assumptions are not satisfied. In all tests, the significance level is set at 0.05. The p-value when comparing all eleven configurations is less than 0.0000, meaning there are significant differences in performance. We hypothesise that the configurations that combine different neighbourhoods are better than each of the configurations with standalone neighbourhoods. Therefore, two additional tests are executed. First, a Kruskal–Wallis test is used to check whether there

Table 4
Results for the different (combinations of) neighbourhoods in the VNDS heuristic. For each configuration and dataset, the results listed here are the average of three algorithm runs with different seeds. The available computation time per run was limited to 1 h.

Configuration	Probabilities (pct.)					Results				
	$\mathcal{N}_2^D$	$\mathcal{N}_3^D$	$\mathcal{N}_4^D$	$\mathcal{N}^S$	$\mathcal{N}_{10}^T$	INEM	INEM MP	Test03	Test05	Test10
1	100	0	0	0	0	27,416	41,341	54,123	27,807	139,909
2	0	100	0	0	0	27,177	41,038	53,319	27,615	136,228
3	0	0	100	0	0	27,108	41,177	53,243	27,594	459,961
4	0	0	0	100	0	27,498	41,326	60,966	28,140	27,381
5	0	0	0	0	100	382,520	2,022,310	82,382	43,850	3,019,970
6	40	30	20	8	2	27,024	40,797	53,113	27,465	25,604
7	35	27.5	17.5	15	5	27,037	40,795	53,233	27,487	25,719
8	35	27.5	17.5	10	10	27,031	40,844	52,989	27,497	25,752
9	30	25	15	25	5	27,024	40,855	53,313	27,501	25,838
10	30	25	15	20	10	27,009	40,841	53,303	27,502	25,883
11	30	25	15	15	15	27,026	40,851	53,226	27,500	25,928

are significant differences in performance between configurations (6), (7), (8), (9), (10), and (11). This test yields a p-value of 0.9901 meaning all six configurations have a similar performance. Next, we compare the performance of each of the standalone neighbourhoods with the performance of configuration (6) using non-parametric Wilcoxon signed-rank tests. A Bonferroni correction is used to avoid type I error inflation. All five tests are significant, meaning configuration (6) achieves significantly better results than each of the standalone neighbourhoods. Based on these tests, two conclusions can be drawn. First, it is clear that combining the different neighbourhoods increases performance and makes the algorithm more robust. Indeed, for the Test10 instance, the individual $\mathcal{N}_d^D$ neighbourhoods perform poorly, while the configurations consisting of all neighbourhoods do not suffer from this problem. Second, the concrete probabilities of the different neighbourhoods do not have a significant impact on the performance, which is positive as it makes the heuristic quite robust to its parameter settings. Based on these tests, we choose the probabilities of configuration (6) in all subsequent tests.

5.4. Results

First, CPLEX was used to solve the INEM instance. If the computation time is limited to only 1 h, CPLEX performs very poorly with a best found solution of 1,994,304 (corresponding to a 98.67 percent optimality gap). By contrast, both the diving heuristic and the VNDS heuristic succeed in finding significantly better solutions with objective values of 107,444 (75.39 percent optimality gap) for the diving heuristic where in each iteration of the column generation phase a subproblem is solved for each person, 76,556 (65.46 percent optimality gap) for the diving heuristic where during the column generation phase the master is reoptimised after each addition of a new column, and an average of 27,024 (2.15 percent optimality gap) for the VNDS heuristic respectively. Moreover, even if CPLEX is given an available computation time of 5 hours, its best found solution with a value of 29,130 (corresponding to a 9.23 percent optimality gap), is still significantly worse than the solution found by the VNDS heuristic in only 1 h. These results illustrate the practical value of using a heuristic approach in practice.

The results for the diving heuristic with both column generation schemes and the VNDS heuristic are provided in Table 5. Because the VNDS heuristic is stochastic, we take the average results over three algorithm runs. Three runs were deemed sufficient since the standard deviation between the different runs was rather small (i.e., an average coefficient of variation of only 0.26 percent). The results of both the LP relaxation and the IP model of Eqs. (1)–(12) solved by CPLEX are also presented. The average solution time of the LP relaxation equals 945 seconds, with a maximum of 3254 s for the Test11 instance. These numbers illustrate that the problem instances are hard to solve. From the results of Table 5, three observations can be made. First, all three heuristics outperform CPLEX within the available computation time of only 1 h. Second, the column generation scheme has a clear impact on the performance of the diving heuristic. The scheme where the master is reoptimised each time a new column with negative reduced cost has been found achieves notably better results than solving a subproblem for every person during each iteration. Third, for our problem, both diving heuristic implementations perform poorly relative to the VNDS heuristic, which achieves very good results. For 10 instances of the 20 instances, the VNDS heuristics finds solutions with an optimality gap of under 3 percent, while for another 4 instances the gap is under 5 percent. For only 3 instances the gap exceeds 10 percent. These results clearly demonstrate the strength of the proposed VNDS heuristic for this type of problem. The small standard deviation between different algorithm runs also shows that the algorithm is quite successful in escaping from bad local optima. Two possible reasons for the poor performance of the diving heuristics are the fact that understaffing and overstaffing are allowed in constraints (19) and the large difference in objective function coefficients, which lead to a slow convergence of the column generation phase.

For the VNDS heuristic, we investigate which factors determine whether an instance is hard to solve. Therefore, a linear regression model is fitted with the optimality gap as dependent variable and four instance characteristics as predictor variables, namely the number of x_{ptds} variables, the LP relaxation solution time, the skill level, and the utilisation. We also include an interaction term for the skill level and utilisation, since a lower utilisation might not make the problem easier if the skill level also decreases. The results are listed in Table 6. The adjusted R^2 of the model equals 0.779, meaning around 78 percent of the variance in the optimality gaps between instances is explained by the four dataset characteristics. Contrary to what one might expect, neither the number of x_{ptds} variables nor the LP relaxation solution time have a significant impact on the performance of the VNDS heuristic. On the other hand, when taken together, the skill level and utilisation are good predictors of the difficulty of a problem instance. As the utilisation increases and the skill level decreases, the performance of the VNDS heuristic decreases. This can be observed for the Test02, Test03, and Test04 instances, which are the only ones with gaps above 10 percent. These instances simultaneously have high utilisations of 135, 129, and 134 percent respectively and low skill levels of only 52, 33, and 51 percent respectively. However, we have to be careful in generalising conclusions from this limited experiment consisting of only 20 instances.

Table 5
Results. 'Diving A' denotes the diving heuristic with the first column generation scheme where for every person a subproblem is solved in each iteration. 'Diving B' denotes the diving heuristic with the second column generation scheme where the master is reoptimised each time a new column is added. For the VNDS heuristic, 'avg. obj.' and 'SD obj.' are the average and the standard deviation of the objective values of three algorithm runs with different seeds. The average gap is calculated as $1 - \frac{\text{LP opt.}}{\text{avg. obj.}}$ and expressed as a percentage. For the IP model, the diving heuristic, as well as the VNDS heuristic, the available computation time was limited to 1 h.

Instance	LP relaxation		CPLEX IP		Diving A		Diving B		VNDS		
	Time (s)	Opt.	Obj.	Gap	Obj.	Gap	Obj.	Gap	Avg. obj.	SD obj.	Avg. gap
INEM	180	26,442	1,994,304	98.67	107,444	75.39	76,556	65.46	27,024	42	2.15
INEM MD	572	50,506	2,255,012	97.76	358,032	85.89	367,952	86.27	51,488	55	1.91
INEM MP	272	39,218	2,821,106	98.53	189,018	79.25	100,314	60.91	40,797	16	3.87
INEM LS	372	41,386	3,698,754	98.94	228,692	81.90	191,966	78.44	42,452	136	2.51
INEM HS	304	25,128	441,896	94.31	263,890	90.48	100,720	75.05	26,062	351	3.58
Test01	505	19,892	253,624	92.16	80,406	75.26	58,638	66.08	20,389	20	2.44
Test02	1557	61,949	737,112	91.60	516,696	88.01	298,520	79.25	74,090	6	16.39
Test03	638	47,486	417,742	88.63	480,520	90.12	214,088	77.82	53,113	191	10.59
Test04	1227	55,322	557,332	90.07	597.340	90.74	307,248	81.99	61,845	7	10.55
Test05	829	26,711	251,460	89.38	202,948	86.84	81,446	67.20	27,465	31	2.74
Test06	375	15,140	3,887,199	99.61	61,235	75.28	32,247	53.05	15,542	4	2.59
Test07	1596	18,337	1,251,654	98.53	514,448	96.44	125,758	85.42	18,799	115	2.46
Test08	251	10,386	89,688	88.42	54,612	80.98	41,824	75.17	10,462	0	0.73
Test09	1852	38,398	1,251,960	96.93	240,842	84.06	121,378	68.36	41,258	21	6.93
Test10	2573	25,120	5,157,815	99.51	267,577	90.61	110,027	77.17	25,604	195	1.89
Test11	3254	22,065	1,454,516	98.48	434,378	94.92	144,192	84.70	23,269	20	5.18
Test12	689	15,742	4,022,071	99.61	126,017	87.51	45,007	65.02	16,501	23	4.60
Test13	563	6452	883,480	99.27	164,252	96.07	68,164	90.53	6944	66	7.09
Test14	626	14,193	4,580,454	99.69	120,256	88.20	52,846	73.14	14,625	4	2.95
Test15	659	10,944	2,581,719	99.58	92,371	88.15	42,821	74.44	11,408	7	4.07

Table 6
Results of the regression model.

Factor	Estimate	Standard error	*t*-statistic	*p*-value
(Intercept)	−0.3421	0.0511	−6.6904	0.0000
Number of x_{ptds} variables	0.0000	0.0000	−0.1281	0.8985
LP relaxation solution time	0.0000	0.0000	−0.9187	0.3623
Skill level	0.3109	0.0631	4.9320	0.0000
Utilisation	0.3761	0.0440	8.5440	0.0000
Skill level × Utilisation	−0.2844	0.0666	−4.2717	0.0000

5.5. Validation and implementation

In this section, the solutions proposed by the VNDS heuristic are compared with the actual schedule used by INEM. The VNDS is implemented, since it clearly outperforms both diving heuristics. The impact of different objective function weights on the proposed schedule is explored. Furthermore, two what-if scenarios are investigated to show how an expert system can assist in taking managerial decisions.

In a first step, we have implemented the VNDS heuristic in a decision support system with a GUI that aims at achieving three objectives. First, it allows easy data input. The imported data are listed in a tree view so that they can be validated by the user. Second, it visualises the found solution in two different ways. A first tab shows the tasks assigned to each person over the planning horizon (see Fig. A.2). During the search, the fixed decisions are indicated in green, while improvements to the solution by one of the neighbourhoods are indicated in blue. A second tab shows for each task the demand in each period and the number of workers that are assigned (see Fig. A.3). If there is understaffing, the corresponding cells are indicated in red. Conversely, overstaffing is indicated in blue. Third, both the objective function weights as well as the allowed computation time can easily be changed in the settings menu (see Fig. A.4). This expert system helps in illustrating how the algorithm works and in visualising the proposed solution to the different stakeholders involved in the project.

In what follows, we compare the proposed schedule with the actual schedule that was implemented by INEM. In this actual schedule, there are only 278 people listed compared to 289 for the data of the INEM instance that were provided to us initially, while the demand in the actual schedule is 4770 tasks versus 4527 in the INEM instance. Finally, in the implemented schedule there is one holiday meaning the contract hours are 132 instead of 140 as is assumed in the original dataset. To make a fair comparison, we create a new dataset named 'INEM2', with the same data as in the actual schedule that was implemented.

Six different scenarios are considered for the choice of the objective function weights. The first is the base case, which uses the same objective function weights as in the computational tests of Sections 5.3 and 5.4. These weights were chosen based on the stated importance of the different objectives, taking into account the different dimensions in which they are measured. The second scenario attributes a higher importance to the working time objective. The third scenario gives a higher focus to assigning tasks to people within their own team. The fourth scenario considers full weekends off more important. The fifth scenario lowers the penalty for understaffing in the EVs and increases the penalties for both overtime and undertime, but with a higher relative cost for overtime. Finally, the sixth scenario explores the impact of giving all objective terms the same weights. All scenarios with the respective objective function weights are shown in Table 7.

The performance on four dimensions is compared: the demand coverage, the actual working time compared to the contract hours, the number of full weekends off and the percentage of tasks that are assigned to members within their own teams. For the first dimension, the total amount of understaffing and overstaffing are calculated. For the second dimension, we look at the average working hours over the planning horizon per worker, the average shortage in hours worked per worker ('undertime'), and the average number of hours of overtime per worker. This gives a total of eight

Table 7
Objective function weights used in the different scenarios.

Scenario	w^{RE+}	w^{RE-}_{CODU}	w^{RE-}_{EV}	w^{WO}	w^{H+}	w^{H-}	w^{G}_{CODU}	w^{G}_{EV}
1. Base case	10	100	1000	10	1	1	10	20
2. Higher focus on working time	10	100	1000	10	10	10	10	20
3. Higher focus on tasks within group	10	100	1000	1	1	10	50	100
4. Higher focus on full weekends off	10	100	1000	1	1	200	10	20
5. Understaffing less, overtime more costly	10	100	100	15	5	10	10	20
6. All weights equal	100	100	100	100	100	100	100	100

Table 8
Comparison of the schedules obtained by the VNDS with the real schedule implemented by INEM. US denotes total understaffing over the planning horizon, i.e. $\sum_{tds} Y^{RE-}_{tds}$. OS denotes total overstaffing over the planning horizon, i.e. $\sum_{tds} Y^{RE+}_{tds}$. HW denotes the average number of hours worked per person over the planning horizon. UT denotes the average shortage in hours worked over all workers, i.e. $\sum_p Y^{H-}_p/|P|$. OT denotes the average excess in hours worked over all workers, i.e. $\sum_p Y^{H+}_p/|P|$. WO denotes the average number of full weekends off per person. Finally, TWT denotes the percentage of tasks that are done by someone of the team to which the task is assigned. For the schedule implemented by INEM, no data on the UT and OT were available, which is indicated by "n/a" in the corresponding cells.

Instance	Scenario	US	OS	HW	UT	OT	WO	TWT
	Target	0	0	132	0	0	4	100
INEM2	Schedule implemented by INEM	727	26	117	n/a	n/a	0.99	83.40
INEM2	Base case	0	0	140	−3.15	10,78	1.44	92.75
INEM2	Higher focus on working time	0	0	140	0,00	7.63	1.46	91.64
INEM2	Higher focus on tasks within group	0	0	140	−4.99	12.62	1.33	92.62
INEM2	Higher focus on weekends off	0	12	140	−3.02	10.99	1.51	91.99
INEM2	Understaffing less, overtime more costly	25	0	137	0.00	4.86	1.56	91.88
INEM2	All weights equal	46	6	137	0.00	5.18	1.58	92.02
INEM2 HS	Understaffing less, overtime more costly	31	0	137	0.00	4.60	1.57	91.98
INEM	Understaffing less, overtime more costly	0	0	128	−5.11	1.13	1.64	85.09

KPIs to compare the different schedules. Table 8 gives the results. The actual schedule used by INEM fails to meet 727 of 4770 demands for tasks over the entire planning horizon. At the same time, however, on 26 occasions more people are assigned than necessary. By contrast, the base case schedule proposed by the VNDS heuristic perfectly meets the demand requirements. Next, people work only 117 hours on average in the actual schedule, considerably below their target of 132 hours. In the base case schedule of the VNDS heuristic on the other hand, the average working time equals 140 hours, which means that the schedule uses overtime (an average of 1 shift per person) to meet all demand requirements. Furthermore, in the proposed schedule people receive on average 1.44 full weekends off compared to only 0.99 weekends in the actual schedule. Finally, the number of tasks assigned to members within the team is also higher in the proposed schedule (91.64%) than in the schedule implemented in practice (83.40%). Therefore, we can conclude that the VNDS heuristic is a clear improvement over the time-consuming manual scheduling procedure.

Next, the different scenarios are compared. Increasing the objective function weights for the overtime and 'undertime' gives the same average working time of 140 h as in the base case. However, the average 'undertime' and overtime are considerably smaller, meaning there is less variation in the working time between different people. Since already more than 90 percent of tasks are assigned to members within the team, increasing the objective function weights for this factor does not have an impact on the schedule, as is shown by scenario (3). When the objective function weight for full weekends off is increased, there is a small increase in the number of full weekends off per person from 1.44 to 1.51. When overstaffing in the EVs is penalised less and at the same time overtime is more costly, the variation in the working time between different people decreases, with only 4.86 hours overtime on average. Also the number of full weekends off increases due to the fact that less overtime is used. However, this comes at the cost of 25 demands that could not be met. Finally, when all objective function weights are given the same value, understaffing and overstaffing increase further, because their relative importance decreases.

Both the base case schedule as well as the schedule of scenario (5) (with understaffing valued less and overtime valued more) score well on different KPIs. Which one is best depends on the preference of the organisation.

The decision support system can also be used to test the impact of certain management decisions. We consider two scenarios. A first scenario checks the value of giving workers training so that people are qualified to perform more types of tasks. To illustrate this the INEM2 dataset is adapted to the extreme case where every person would have the required skills for every task. This dataset is referred to as INEM2 HS. A second scenario compares the schedules found for the INEM2 instance with the schedules found for the INEM instance, where an additional 11 people are available to meet the demand requirements. We test both scenarios with the objective function weights used in scenario (5). The results are listed in the bottom of Table 8. The results show that increasing training in this case does not improve the quality of the schedule. On the other hand, increasing the available workforce by 11 people enables us to meet all demand requirements, with less overtime and more full weekends off. Of course, the average working time decreases and the average worker works 5.11 hours less than their contract hours. The organisation can use the decision support system to quickly carry out various what-if scenarios.

6. Conclusion

This paper addresses a real-life personnel scheduling problem at a medical emergency service. First, the problem is formulated as an integer program. Because the integer programming formulation turns out to be intractable for real-life problem instances, a diving heuristic and a VNDS heuristic are developed. In the diving heuristic, the LP relaxation is reformulated by decomposing on the staff members and solved using column generation. Integer solutions are obtained by heuristically branching on all variables with a value above a certain threshold until for each person the schedule has been fixed. Moreover, two different column generation schemes are compared. The VNDS heuristic combines the principles of fix-and-optimise and VNDS. The fix-and-optimise heuristic uses a MIP solver to solve a sequence of subproblems in which only a subset of the variables can be changed, while all other variables are fixed to the value of the current solution. This fix-and-optimise method is embedded within the VNDS framework, which controls the search. A local search phase iteratively improves the current solution by combining different neighbourhood structures, while a shake phase is used to escape local optima.

Both heuristics are extensively tested on a real-life case study at Instituto Nacional de Emergência Médica (INEM) as well as a set of nineteen instances of different dimensions. Four instances are derived from the INEM dataset by changing one of the problem dimensions. A random instance generator is developed to construct fifteen additional instances. The MIP-heuristic significantly outperforms both diving heuristic implementations. It achieves good results with an average optimality gap across all instances of 4.76 percent in only one hour of computation time. For the MIP-heuristic, computational experiments show that the most important factor that contributes to the difficulty of a problem instance is the utilisation level (i.e., the total demand for tasks divided by the available number of employees). Contrary to what one might expect, there is no significant correlation between the performance of the MIP-heuristic and the size of the problem instances as measured by the number of decision variables.

The solution provided by the VNDS heuristic is also compared with the actual schedule implemented by INEM. To facilitate the application of the algorithm in practice and illustrate its usefulness to management, the VNDS heuristic has been implemented in an expert system with a GUI that visualises the search process and the found solution. Six schedules proposed by the expert system using different objective function weights are compared and evaluated on eight KPIs. All schedules outperform the actual schedule implemented by INEM. Also two what-if scenarios are investigated to demonstrate how the expert system can help the organisation in making managerial decisions.

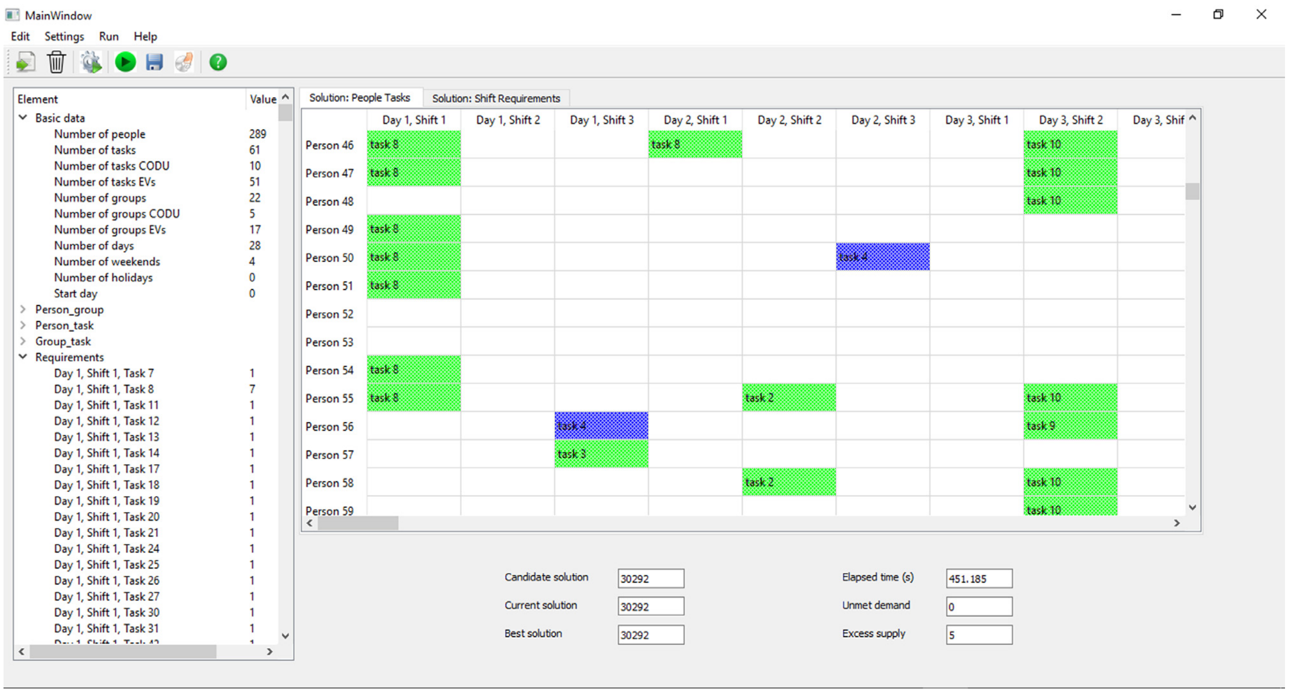
While different solution approaches are provided in the literature, developing an expert system for real-life staff scheduling problems based on these theoretical concepts is not straightforward. Indeed, while diving heuristics perform well on the NSP, for our problem setting they proved ineffective. Furthermore, for the VNDS heuristic, the neighbourhood decompositions used in the local search phase need to be tailored to the problem formulation and different parameter configurations need to be tested.

Finally, we provide some ideas for future research. First, it would be interesting to extend the current scheduling tool to take care of holidays. People could specify certain days off they would like to receive and the algorithm could try to respect these preferences as much as possible. As a second idea, a more in-depth benchmarking of different schedules proposed by the heuristic could be carried out. One such possibility is the use of data envelopment analysis to see which schedules are on the efficient frontier based on a given set of KPIs (see, e.g., Van den Bergh, De Bruecker, Beliën, De Boeck, & Demeulemeester, 2013). Third, an efficient scheduling tool for our problem is now available, but it might still be difficult to understand and use for people not familiar with operations research. For example, the input data need to be provided in the specific format that the algorithm can use. Furthermore, understanding how the algorithm works and what its limitations are is important for the expectations of the user. The question is thus how to ensure that the layperson can work effectively with the decision support system. Finally, another future research direction could be the design of methods to deal with rescheduling in the course of the planning period. A methodology may be developed and embedded in the decision support system.

Acknowledgments

We would like to thank Jaime Naia Teixeira of Instituto Nacional de Emergência Médica for the support in validating the model and providing the data for the case study. We would also like to thank Bernardo Almada-Lobo for helpful suggestions in improving this paper. Hendrik Vermuyten was sponsored by the Research Foundation Flanders (FWO), travel grant no. V418317N.

Appendix A. Graphical user interface

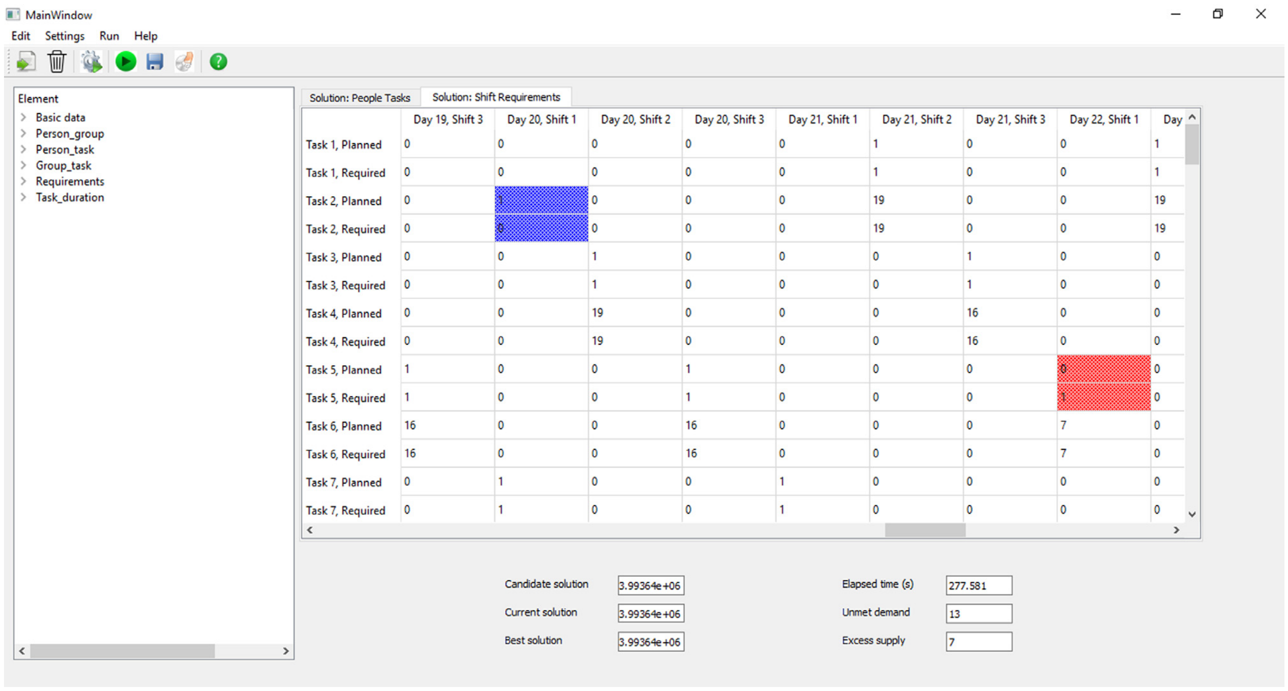


Element	Value
Basic data	
Number of people	289
Number of tasks	61
Number of tasks CODU	10
Number of tasks EVs	51
Number of groups	22
Number of groups CODU	5
Number of groups EVs	17
Number of days	28
Number of weekends	4
Number of holidays	0
Start day	0
Person_group	
Person_task	
Group_task	
Requirements	
Day 1, Shift 1, Task 7	1
Day 1, Shift 1, Task 8	7
Day 1, Shift 1, Task 11	1
Day 1, Shift 1, Task 12	1
Day 1, Shift 1, Task 13	1
Day 1, Shift 1, Task 14	1
Day 1, Shift 1, Task 17	1
Day 1, Shift 1, Task 18	1
Day 1, Shift 1, Task 19	1
Day 1, Shift 1, Task 20	1
Day 1, Shift 1, Task 21	1
Day 1, Shift 1, Task 24	1
Day 1, Shift 1, Task 25	1
Day 1, Shift 1, Task 26	1
Day 1, Shift 1, Task 27	1
Day 1, Shift 1, Task 30	1
Day 1, Shift 1, Task 31	1

	Day 1, Shift 1	Day 1, Shift 2	Day 1, Shift 3	Day 2, Shift 1	Day 2, Shift 2	Day 2, Shift 3	Day 3, Shift 1	Day 3, Shift 2	Day 3, Shif
Person 46	task 8			task 8				task 10	
Person 47	task 8							task 10	
Person 48								task 10	
Person 49	task 8								
Person 50	task 8					task 4			
Person 51	task 8								
Person 52									
Person 53									
Person 54	task 8								
Person 55	task 8				task 2			task 10	
Person 56			task 4					task 9	
Person 57			task 3						
Person 58					task 2			task 10	
Person 59								task 10	

Candidate solution 30292
Current solution 30292
Best solution 30292
Elapsed time (s) 451.185
Unmet demand 0
Excess supply 5

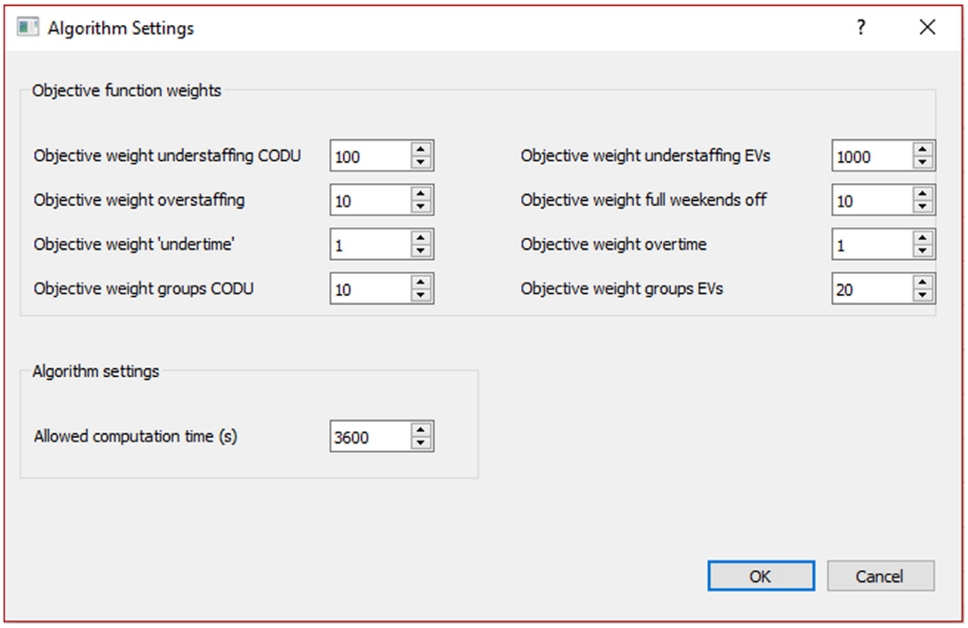
Fig. A.2. Graphical user interface: visualisation of the current best solution. This view shows the tasks assigned to each person over the planning horizon. The blue cells indicate the new assignments obtained by the neighbourhood, while the green cells indicate decisions that were fixed. In the left tree view, the problem data are shown so that the user can check whether all data are correct. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)



	Day 19, Shift 3	Day 20, Shift 1	Day 20, Shift 2	Day 20, Shift 3	Day 21, Shift 1	Day 21, Shift 2	Day 21, Shift 3	Day 22, Shift 1	Day
Task 1, Planned	0	0	0	0	0	1	0	0	1
Task 1, Required	0	0	0	0	0	1	0	0	1
Task 2, Planned	0	1	0	0	0	19	0	0	19
Task 2, Required	0	0	0	0	0	19	0	0	19
Task 3, Planned	0	0	1	0	0	0	1	0	0
Task 3, Required	0	0	1	0	0	0	1	0	0
Task 4, Planned	0	0	19	0	0	0	16	0	0
Task 4, Required	0	0	19	0	0	0	16	0	0
Task 5, Planned	1	0	0	1	0	0	0	0	0
Task 5, Required	1	0	0	1	0	0	0	1	0
Task 6, Planned	16	0	0	16	0	0	0	7	0
Task 6, Required	16	0	0	16	0	0	0	7	0
Task 7, Planned	0	1	0	0	1	0	0	0	0
Task 7, Required	0	1	0	0	1	0	0	0	0

Candidate solution	3.99364e+06	Elapsed time (s)	277.581
Current solution	3.99364e+06	Unmet demand	13
Best solution	3.99364e+06	Excess supply	7

Fig. A.3. Graphical user interface: visualisation of the current best solution. This view shows the supply and demand for each task over the planning horizon. Red cells indicate understaffing, while blue cells indicate overstaffing. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)



Objective function weights

Objective weight understaffing CODU	100	Objective weight understaffing EVs	1000
Objective weight overstaffing	10	Objective weight full weekends off	10
Objective weight 'undertime'	1	Objective weight overtime	1
Objective weight groups CODU	10	Objective weight groups EVs	20

Algorithm settings

Allowed computation time (s) 3600

Fig. A.4. Graphical user interface: settings dialog.

References

Addis, B., Aringhieri, R., Carello, G., Grosso, A., & Maffioli, F. (2012). Workforce management based on forecasted demand. In E. Tànfani, & A. Testi (Eds.), *Advanced decision making methods applied to health care* (pp. 1–11). Milano: Springer Milan. doi:10.1007/978-88-470-2321-5_1.

Aickelin, U., & Dowsland, K. A. (2000). Exploiting problem structure in a genetic algorithm approach to a nurse rostering problem.. *Journal of Scheduling, 3*(3), 139–153. doi:10.1002/(SICI)1099-1425(200005/06)3:3⟨139::AID-JOS41⟩3.0.CO;2-2.

Aickelin, U., & Dowsland, K. A. (2004). An indirect genetic algorithm for a nurse-scheduling problem. *Computers and Operations Research, 31*(5), 761–778. doi:10.1016/S0305-0548(03)00034-0.

Altamirano, L., Riff, M. C., Araya, I., & Trilling, L. (2012). Anesthesiology nurse scheduling using particle swarm optimization. *International Journal of Computational Intelligence Systems, 5*(1), 111–125. doi:10.1080/18756891.2012.670525.

Aringhieri, R., Bruni, M. E., Khodaparasti, S., & van Essen, J. T. (2017). Emergency medical services and beyond: Addressing new challenges through a wide literature review. *Computers and Operations Research, 78*(August 2016), 349–368. doi:10.1016/j.cor.2016.09.016.

Bard, J. F., Binici, C., & DeSilva, A. H. (2003). Staff scheduling at the United States Postal Service. *Computers and Operations Research, 30*(5), 745–771. doi:10.1016/S0305-0548(02)00048-5.

Bard, J. F., & Purnomo, H. W. (2005a). A column generation-based approach to solve the preference scheduling problem for nurses with downgrading. *Socio-Economic Planning Sciences, 39*(3), 193–213. doi:10.1016/j.seps.2004.04.001.

Bard, J. F., & Purnomo, H. W. (2005b). Preference scheduling for nurses using column generation. *European Journal of Operational Research, 164*(2), 510–534. doi:10.1016/j.ejor.2003.06.046.

Başar, A., Çatay, B., & Ünlüyurt, T. (2012). A taxonomy for emergency service station location problem. *Optimization Letters, 6*(6), 1147–1160. doi:10.1007/s11590-011-0376-1.

Beliën, J., & Demeulemeester, E. (2006). Scheduling trainees at a hospital department using a branch-and-price approach. *European Journal of Operational Research, 175*(1), 258–278. doi:10.1016/j.ejor.2005.04.028.

Bellanti, F., Carello, G., Della Croce, F., & Tadei, R. (2004). A greedy-based neighborhood search approach to a nurse rostering problem. *European Journal of Operational Research, 153*(1), 28–40. doi:10.1016/S0377-2217(03)00096-1.

Belo-Filho, M., Amorim, P., & Almada-Lobo, B. (2015). An adaptive large neighbourhood search for the operational integrated production and distribution problem of perishable products. *International Journal of Production Research, 53*(20), 6040–6058. doi:10.1080/00207543.2015.1010744.

Bradbeer, P. V. G., Findlay, C., & Fogarty, T. (2000). An ambulance crew rostering system. In S. Cagnoni (Ed.), *Proceedings of the Real-world applications of evolutionary computing: Evoworkshops: Evoiasp, evoscondi, evotel, evostim, evorob, and evoflight Edinburgh, Scotland, UK, April 17* (pp. 267–279). Berlin, Heidelberg: Springer Berlin Heidelberg. doi:10.1007/3-540-45561-2_26.

Brucker, P., Qu, R., & Burke, E. (2011). Personnel scheduling: Models and complexity. *European Journal of Operational Research, 210*(3), 467–473. doi:10.1016/j.ejor.2010.11.017.

Burke, E., Causmaecker, P., Vanden Berghe, G., & Landeghem, H. V. (2004). The state of the art of nurse scheduling. *Journal of Scheduling, 7*(6), 441–499.

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

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

Burke, E. K. E., & Curtois, T. (2010). An ejection chain method and a branch and price algorithm applied to the instances of the first international nurse rostering competition, 2010, *10*, 13.

Camargo, V., Toledo, F., & Almada-Lobo, B. (2014). HOPS – Hamming-Oriented Partition Search for production planning in the spinning industry. *European Journal of Operational Research, 234*(1), 266–277. doi:10.1016/j.ejor.2013.10.017.

Cheang, B., Li, H., Lim, A., & Rodrigues, B. (2003). Nurse rostering problems – A bibliographic survey. *European Journal of Operational Research, 151*(3), 447–460. doi:10.1016/S0377-2217(03)00021-3.

Cline, D., Reilly, C., & Moore, J. F. (2003). What's behind RN turnover? *Nursing management, 34*(10), 50–53. doi:10.1097/00006247-200310000-00016.

Dantzig, G. B., & Wolfe, P. (1960). Decomposition principle for linear programs. *Operations Research, 8*(1), 101–111.

Dorneles, Á. P., De Araújo, O. C. B., & Buriol, L. S. (2014). A fix-and-optimize heuristic for the high school timetabling problem. *Computers and Operations Research, 52*(PART A), 29–38. doi:10.1016/j.cor.2014.06.023.

Erdoan, G., Erkut, E., Ingolfsson, A., & Laporte, G. (2010). Scheduling ambulance crews for maximum coverage. *The Journal of the Operational Research Society, 61*(4), 543–550. doi:10.2307/40608177.

Erhard, M., Schoenfelder, J., Fügener, A., & Brunner, J. (2017). State of the Art in physician scheduling. *European Journal of Operational Research, 0*, 1–35. doi:10.1016/j.ejor.2017.06.037.

Ernst, A., Jiang, H., Krishnamoorthy, M., Owens, B., & Sier, D. (2004a). An annotated bibliography of personnel scheduling and rostering. *Annals of Operations Research, 127*, 21–144. doi:10.1023/B:ANOR.0000019087.46656.e2.

Ernst, A. T., Jiang, H., Krishnamoorthy, M., & Sier, D. (2004b). Staff scheduling and rostering: A review of applications, methods and models. *European Journal of Operational Research, 153*(1), 3–27. doi:10.1016/S0377-2217(03)00095-X.

Gamache, M., Soumis, F., Marquis, G., & Desrosiers, J. (1999). A column generation approach for large-scale aircrew rostering problems. *Operations Research, 47*(2), 247–263. doi:10.1287/opre.47.2.247.

Gomes, R. A., Toffolo, T. A., & Santos, H. G. (2017). Variable neighborhood search accelerated column generation for the nurse rostering problem. *Electronic Notes in Discrete Mathematics, 58*, 31–38. doi:10.1016/j.endm.2017.03.005.

Hans, E. W. (2001). *Resource loading by branch-and-price techniques* (Ph.D. thesis)..

Hansen, P., & Mladenović, N. (2001). Variable neighborhood search: Principles and applications. *European Journal of Operational Research, 130*(3), 449–467.

Hansen, P., Mladenović, N., & Perez-Britos, D. (2001). Variable Neighborhood Decomposition Search. *Journal of Heuristics, 7*(4), 335–350.

Helber, S., & Sahling, F. (2010). A fix-and-optimize approach for the multi-level capacitated lot sizing problem. *International Journal of Production Economics, 123*(2), 247–256. doi:10.1016/j.ijpe.2009.08.022.

Henderson, S. (2011). Operations research tools for addressing current challenges in emergency medical services. *Wiley encyclopedia of operations research and management science*. Hoboken, NJ: John Wiley & Sons.

INEM (2017). Missão, Visão e Valores. http://www.inem.pt/PageGen.aspx?WMCM_PaginaId=27724 Accessed: 22 April 2017.

Ingolfsson, A. (2013). EMS planning and management. In G. S. Zaric (Ed.), *Operations research and health care policy* (pp. 105–128). New York, NY: Springer New York. doi:10.1007/978-1-4614-6507-2_6.

Isken, M. W. (2004). An implicit tour scheduling model with applications in healthcare. *Annals of Operations Research, 128*(1–4), 91–109. doi:10.1023/B:ANOR.0000019100.08333.a7.

James, R., & Almada-Lobo, B. (2011). Single and parallel machine capacitated lotsizing and scheduling: New iterative MIP-based neighborhood search heuristics. *Computers and Operations Research, 38*(12), 1816–1825. doi:10.1016/j.cor.2011.02.005.

Joncour, C., Michel, S., Sadykov, R., Sverdlov, D., & Vanderbeck, F. (2010). Column generation based primal heuristics. *Electronic Notes in Discrete Mathematics, 36*(C), 695–702. doi:10.1016/j.endm.2010.05.088.

Kreeft, D. (2012). Development and implementation of a computer-aided method for planning resident shifts in a university hospital. (Master's thesis). Université Laval, Québec.

Li, X., Zhao, Z., Zhu, X., & Wyatt, T. (2011). Covering models and optimization techniques for emergency response facility location and planning: a review. *Mathematical Methods of Operations Research, 74*(3), 281–310. doi:10.1007/s00186-011-0363-4.

Li, Y., & Kozan, E. (2009). Rostering ambulance services. In *Proceedings of the ninth Asia-Pacific industrial engineering and management society, Kitakyushu, Japan* (pp. 795–801).

Ministério da Saúde (1981). Decreto-lei 176/1981, Diário da República, Série I (3 August 1981), Ministério da Saúde.

Moz, M., & Vaz Pato, M. (2007). A genetic algorithm approach to a nurse rerostering problem. *Computers and Operations Research, 34*(3), 667–691. doi:10.1016/j.cor.2005.03.019.

Pato, M. V., & Moz, M. (2008). Solving a bi-objective nurse rerostering problem by using a utopic Pareto genetic heuristic. *Journal of Heuristics, 14*(4), 359–374. doi:10.1007/s10732-007-9040-4.

Puente, J., Gómez, A., Fernández, I., & Priore, P. (2009). Medical doctor rostering problem in a hospital emergency department by means of genetic algorithms. *Computers and Industrial Engineering, 56*(4), 1232–1242. doi:10.1016/j.cie.2008.07.016.

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

Rajagopalan, H. K., Saydam, C., Sharer, E., & Setzler, H. (2011). Ambulance deployment and shift scheduling: An integrated approach. *Journal of Service Science and Management, 04*(01), 66–78. doi:10.4236/jssm.2011.41010.

Santos, H. G., Toffolo, T. A. M., Gomes, R. A. M., & Ribas, S. (2016). Integer programming techniques for the nurse rostering problem. *Annals of Operations Research, 239*(1), 225–251. doi:10.1007/s10479-014-1594-6.

Seeanner, F., Almada-Lobo, B., & Meyr, H. (2013). Combining the principles of variable neighborhood decomposition search and the fix & optimize heuristic to solve multi-level lot-sizing and scheduling problems. *Computers and Operations Research, 40*(1), 303–317. doi:10.1016/j.cor.2012.07.002.

Smet, P., Bilgin, B., De Causmaecker, P., & Vanden Berghe, G. (2014). Modelling and evaluation issues in nurse rostering. *Annals of Operations Research, 218*(1), 303–326. doi:10.1007/s10479-012-1116-3.

Valouxis, C., Gogos, C., Goulas, G., Alefragis, P., & Housos, E. (2012). A systematic two phase approach for the nurse rostering problem. *European Journal of Operational Research, 219*(2), 425–433. doi:10.1016/j.ejor.2011.12.042.

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

Van den Bergh, J., De Bruecker, P., Beliën, J., De Boeck, L., & Demeulemeester, E. (2013). A three-stage approach for aircraft line maintenance personnel rostering using MIP, discrete event simulation and DEA. *Expert Systems with Applications, 40*(7), 2659–2668. doi:10.1016/j.eswa.2012.11.009.

Vanden Berghe, G. (2013). NURSE ROSTERING: Focus on models or algorithms? [Powerpoint slides]. https://www.utwente.nl/en/choir/events/ArchiveSymposia/CHOIR-practice/VdBerghe20131122.pdf.

Vanderbeck, F. (2000). On Dantzig–Wolfe decomposition in integer programming and ways to perform branching in a branch-and-price algorithm. *Operations Research, 48*(1), 111–128. doi:10.1287/opre.48.1.111.12453.

Vile, J. L., Gillard, J. W., Harper, P. R., & Knight, V. A. (2016). Time-dependent stochastic methods for managing and scheduling emergency medical services. *Operations Research for Health Care, 8*, 42–52. doi:10.1016/j.orhc.2015.07.002.

Zheng, Z., Liu, X., & Gong, X. (2017). A simple randomized variable neighbourhood search for nurse rostering. *Computers and Industrial Engineering, 110*, 165–174. doi:10.1016/j.cie.2017.05.027.