European Journal of Operational Research 249 (2016) 67–75

Contents lists available at ScienceDirect

European Journal of Operational Research

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

Discrete Optimization

Polynomially solvable personnel rostering problems

Pieter Smet*, Peter Brucker[1], Patrick De Causmaecker, Greet Vanden Berghe

KU Leuven, Department of Computer Science, CODeS & iMinds-ITEC, Gebroeders De Smetstraat 1, 9000 Gent, Belgium

ARTICLE INFO

Article history:
Received 8 September 2014
Accepted 17 August 2015
Available online 22 August 2015

Keywords:
Personnel rostering
Polynomial time algorithms
Assignment
Networks

ABSTRACT

Personnel rostering is a personnel scheduling problem in which shifts are assigned to employees, subject to complex organisational and contractual time-related constraints. Academic advances in this domain mainly focus on solving specific variants of this problem using intricate exact or (meta)heuristic algorithms, while little attention has been devoted to studying the underlying structure of the problems. The general assumption is that these problems, even in their most simplified form, are NP-hard. However, such claims are rarely supported with a proof for the problem under study. The present paper refutes this assumption by presenting minimum cost network flow formulations for several personnel rostering problems. Additionally, these problems are situated among the existing academic literature to obtain insights into what makes personnel rostering hard.

1. Introduction

Research on personnel rostering, and personnel scheduling in general, has focused mostly on solving some problem at hand. As a result, a large part of the academic literature details algorithms tailored to one specific problem. Typically, general complexity claims are made, thereby referring to NP-complete problems that *resemble* the problem under discussion. However, in many cases there is no certainty that these complexity claims hold for this particular rostering problem. Theoretical studies on models and complexity of personnel rostering are lacking in the present literature.

There are only a few authors who have formally determined the hardness of a personnel rostering problem. Osogami and Imai (2000) and Brunner, Bard, and Köhler (2013) prove that rostering problems with constraints on the number of assignments of particular shifts, and with constraints on consecutive days worked and days-off are hard. Lau (1996b) describes a shift assignment problem closely related to rostering, and proves its NP-completeness. For restricted variants of the problem, Lau (1996a; 1996b) provide polynomial time algorithms, which are discussed in detail in Section 4.2 of the present paper.

To the best of our knowledge, Brucker, Qu, and Burke (2011) are the only authors to systematically study personnel scheduling from a theoretical point of view. Based on a general mathematical model, four polynomially solvable cases have been identified, two of which are closely related to rostering. The first problem, P_{Thm1}, considers different shifts which require a constant number of employees on different days. The employees are assumed to be available on all days. Without any further restrictions on the assignment of shifts to employees, the problem can be solved as a series of transshipment problems. The second problem, P_{Thm2}, assumes one type of shift, and the availability of employees given by one interval, i.e. employee availability is assumed to be contiguous. There are no other restrictions. A reformulation models the problem as a minimum cost network flow problem.

The present paper identifies new personnel rostering problems that can also be solved in polynomial time. Table 1 compares the two polynomially solvable rostering problems studied by Brucker et al. (2011) with the problems discussed in this work.

In the light of the new contributions, complexity results from the academic literature are revisited to obtain insights into what it is that makes personnel rostering hard. This work provides an update on the current results, and further establishes the foundations for theoretical studies on personnel rostering models.

Even though all results are discussed in terms of *shifts* and *days*, the ideas can be directly transferred to the domain of *tasks* and *periods*. This observation underpins the idea that the presented results have a potential impact not only in different rostering application areas, such as logistics and health care, but also in personnel scheduling in general.

The remainder of this paper is organised as follows. Section 2 introduces basic definitions of concepts in personnel rostering. Section 3 investigates problems with restrictions on the number of

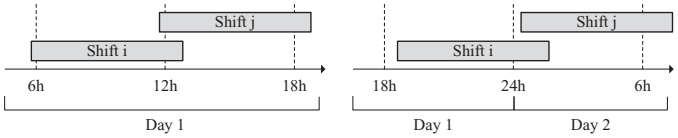
* Corresponding author. Tel.: +3292658704.
E-mail address: pieter.smet@cs.kuleuven.be (P. Smet).

[1] Peter Brucker sadly passed away on July 24, 2013. His coauthors dedicate their contribution in this paper to his memory.

http://dx.doi.org/10.1016/j.ejor.2015.08.025
0377-2217/

Table 1
Comparison of characteristics of polynomially solvable rostering problems from Brucker et al. (2011).

	Shifts		Demand		Employee availability			Time-related constraints
	Single	Multiple	Stable	Varying	Full	Contiguous	Varying	
P_{Thm1}	✓	✓	✓		✓			
P_{Thm2}	✓		✓	✓		✓		
This paper	✓	✓	✓	✓	✓	✓	✓	✓



(a) In-day overlap (b) Next-day overlap

Fig. 1. Types of overlap between shifts.

assignments to each employee. Several polynomially solvable cases are identified by formulating them as minimum cost network flow problems. Based on these results, an efficient approach to a known problem from the literature is presented, and the complexity of commonly used benchmark datasets for nurse rostering is discussed. Sections 4 and 5 consider problems with constraints on consecutive assignments. Again, polynomially solvable cases are presented, and linked with results from the literature. For all results, the practical implications are discussed. Finally, Section 6 concludes the paper and identifies areas for future research.

2. Personnel rostering problems

This section introduces common concepts in personnel rostering problems, which will be used throughout the paper.

Employees have to be assigned to shifts in a way that satisfies a variety of constraints. These problems are characterised by a set of employees $E = \{1, \ldots, e\}$, a scheduling period of days $T = \{1, \ldots, t\}$ and a set of shifts $S = \{1, \ldots, s\}$.

A *shift* is a fixed time interval which denotes a working period. Each shift is characterised by a unique type which classifies the shifts in various ways, e.g., by time interval (morning, late), by required qualifications (senior, junior), or by a combination of these (morning-senior, late-junior). A shift is considered to occur on the day where its time interval starts. The number of employees required for each shift can vary from day to day, and is typically more than one employee.

An *assignment* is the allocation of an employee to a shift on a day. A *roster* is an $e \times t$ matrix which contains in each cell either an assignment or a day-off. If the cardinality of S is one, the single shift represents a day-on, and the solution is referred to as a *day-off roster*. This work considers non-cyclic rosters, in contrast to cyclic rosters in which all employees have the same assignments, but lagged in time (Rocha, Oliveira, & Carravilla, 2013).

Two shifts are *in-day overlapping* if their time intervals overlap when considering the shifts on the same day. An ordered set of two shifts is *next-day overlapping* if an employee cannot be assigned to these shifts on consecutive days without overlap of their time intervals. Fig. 1 visualises these concepts. This distinction is important since several models for personnel rostering problems assume that at most one shift can be assigned per day, thereby automatically eliminating in-day overlap, but not necessarily next-day overlap.

Domain constraints define the possible assignments for each employee on each day. For each employee i and day j, a set of shifts $\bar{S}_{ij}$ is defined, consisting of the shifts that can be feasibly assigned. In practice, these constraints can be used to model restrictions such as 'part-time employees can only work 4 hour or 6 hour shifts' or 'an employee does not want to work late shifts on Wednesday'. This concept can also be used to model employee skills by only including shifts in $\bar{S}_{ij}$ for which employee i is qualified.

The *demand* d_{jk} (or coverage requirement) is the required number of employees on day j, shift k. Demand is *stable* if the same number of employees is required on each day and shift, i.e. $\forall j \in T, k \in S : d_{jk} = d$. Furthermore, if on each day and shift only one employee is required, $\forall j \in T, k \in S : d_{jk} = 1$, there is *unit demand*. In contrast, demand is *varying* if d_{jk} can be any non-negative value.

Demand can be expressed as an *exact*, *ranged*, *minimum* or *maximum* requirement. In the case of exact demand, the specified value is exactly the number of employees to be assigned. A ranged definition requires that the number of assigned employees should be within a specified interval. When such an interval has no upper (lower) limit, the requirement is defined as a minimum (maximum).

In addition to the coverage requirements, personnel rostering problems are typically also subject to a variety of contractual time-related constraints, which can be categorised as *counters*, *series* or *successions*. Counters restrict the number of times a specific roster item (e.g. assignments or days-off) can occur within a certain period. Series restrict consecutive occurrences of specific roster items (Smet, Bilgin, De Causmaecker, & Vanden Berghe, 2014). Similar to the coverage requirements, these different types of constraints can be expressed as either ranged, minimum, maximum or exact. Finally, successions denote a special type of series, which restrict occurrences of specific roster items on two consecutive days.

3. Counter constraints

This section presents results for problems with counter constraints. More specifically, constraints on the number of days worked and on the number of shifts worked of each type are discussed. The literature survey of Van den Bergh, Beliën, De Bruecker, Demeulemeester, and De Boeck (2013) illustrates the importance of these two constraints as they were included in 85 and 47 recent academic studies, respectively.

Each subtitle describes the problem discussed. The first two elements describe the number of shifts and type of demand. The last element states the objective, if any. All other elements describe constraints of the problem. The type of definition (exact, range, minimum or maximum) for each constraint is mentioned between parentheses.

3.1. Single shift, varying demand (minimum), number of days worked (exact), feasibility

The *number of days worked* constraint limits the number of assignments per employee in the scheduling period. In practice, this constraint is used to model different contract types. For example, a full time employee will be required to work 20 days in a monthly scheduling period, whereas a part time employee should only work 15 days.

Consider the set of employees E to be homogeneous, i.e. each employee has to work exactly a days. The problem can be formulated as the following integer linear program.

$$x_{ij} = \begin{cases} 1 & \text{if employee } i \text{ works on day } j \\ 0 & \text{otherwise} \end{cases}$$

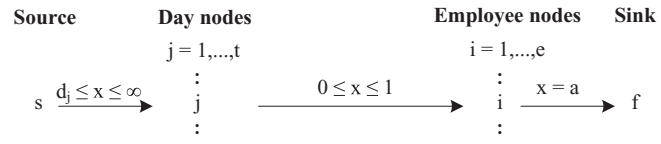


Fig. 2. Network G_1 corresponding with problem $\mathcal{P}_1$. x denotes the flow through an arc.

$$\mathcal{P}_1 : \sum_{i \in E} x_{ij} \geq d_j \quad \forall\, j \in T \tag{1}$$

$$\sum_{j \in T} x_{ij} = a \quad \forall\, i \in E \tag{2}$$

$$x_{ij} \in \{0, 1\} \quad \forall\, i \in E, j \in T \tag{3}$$

Constraints (1) are the coverage requirements. Constraints (2) enforce the number of days worked. Integrality of the decision variables is ensured by constraints (3).

Problem $\mathcal{P}_1$ is a special case of the many-to-many generalised assignment problem (Pentico, 2007), in which tasks need to be assigned to agents. The contribution of each task to an agent's workload is always one. Furthermore, all agents are required to work an exact number of tasks, while for each task only a minimum number of required agents are specified.

3.1.1. Minimum number of employees

Inspired by Burns and Carter (1985), a lower bound on the number of employees required for problem $\mathcal{P}_1$ can be calculated using parameters a and d_j (Equation (4)).

$$e = \max\left(\left\lceil \frac{\sum_{i=1}^{t} d_i}{a} \right\rceil, d_1, d_2, \ldots, d_t\right) \tag{4}$$

The remainder of this section proves that a solution exists with e employees, by applying network flow techniques. First, a layered network is constructed in which a feasible flow corresponds to a solution for problem $\mathcal{P}_1$. Let $G_1 = (V, A)$ be a network with V the set of nodes, and A the set of arcs. The set V consists of four types of nodes: *day nodes* (DN), *employee nodes* (EN), one source node (s), and one sink node (f).

The structure of G_1 is shown in Fig. 2. The supply in all nodes $V \backslash \{s, f\}$ is zero. The supply in the source is ea, which is the maximum number of possible assignments, based on the number of employees and the parameter a. The supply in the sink is $-ea$.

Due to the configuration of G_1, a flow respecting the capacity and demand constraints is equivalent to a solution for problem $\mathcal{P}_1$. One unit of flow in an arc between day node j and employee node i indicates that employee i is working on day j.

Lemma 1. *A feasible flow in the network G_1 corresponds to a feasible solution for problem $\mathcal{P}_1$.*

Based on Lemma 1, there exists a solution with e employees for problem $\mathcal{P}_1$ if a feasible flow exists in G_1. Ahuja, Magnanti, and Orlin (1993) show that the latter can be proven by verifying that the circulation feasibility condition (CFC) holds in network G_1.

Theorem 1. *CFC: A circulation problem with non-negative lower bounds on arc flows is feasible if and only if, for every set N of nodes, with $\bar{N} = V \setminus N$ (Ahuja et al., 1993)*

$$\sum_{(p,q) \in (\bar{N}, N)} l_{pq} \leq \sum_{(p,q) \in (N, \bar{N})} u_{pq} \tag{5}$$

Theorem 2. *There exists a solution for problem $\mathcal{P}_1$ with e employees as calculated by Equation (4).*

Proof. First, G_1 is transformed to a network G_1', representing the corresponding circulation problem. This is done by adding a circulation arc from the sink to the source with infinite positive capacity. Next, it is verified that the CFC holds in network G_1' by checking Equation (5) for nine cases which, due to the network's layered structure, cover all possibilities. Table 2 shows, for each case, which types of nodes are in N and $\bar{N}$, as well as the left and right hand sides of Equation (5). Note that it is assumed that all nodes of a type are in the set for the cases shown in Table 2. Changing this assumption does not change the correctness of the proof, but makes the calculation of the CFC terms more complicated.

Table 2
Cases used in the proof of Theorem 2.

N	$\bar{N}$	$\sum_{(p,q) \in (\bar{N}, N)} l_{pq}$	$\sum_{(p,q) \in (N, \bar{N})} u_{pq}$
$\{s\}$	$\{DN, EN, f\}$	0	$\Sigma_{j \in T} d_j$
$\{s, DN\}$	$\{EN, f\}$	0	et
$\{s, DN, EN\}$	$\{f\}$	0	ea
$\{DN\}$	$\{s, EN, f\}$	$\Sigma_{j \in T} d_j$	et
$\{DN, EN\}$	$\{s, f\}$	$\Sigma_{j \in T} d_j$	ea
$\{DN, EN, f\}$	$\{s\}$	$\Sigma_{j \in T} d_j$	$+\infty$
$\{EN\}$	$\{s, DN, f\}$	0	ea
$\{EN, f\}$	$\{s, DN\}$	0	$+\infty$
$\{f\}$	$\{s, DN, EN\}$	ea	$+\infty$

Table 2 shows that most of the cases are trivial, except the ones with $DN \in N$ and $s \in \bar{N}$. From Equation (4) directly follows that $ea \geq \Sigma_{j \in T} d_j$. Furthermore, since $a \leq t$ always holds, et will always be greater than or equal to $\Sigma_{j \in T} d_j$. It will thus always be possible to construct a feasible flow in G_1'. From Lemma 1 follows that there will always be a solution for problem $\mathcal{P}_1$ with e employees, as calculated by Equation (4). □

3.2. Multiple shifts, varying demand (range), number of days worked (exact), domain constraints, optimise preferences

A common objective in personnel rostering is to respect the employees' preferences as much as possible (Bard & Purnomo, 2005; Vanhoucke & Maenhout, 2007). This is modelled by minimising an integer cost c_{ijk} defined for assigning employee i to shift k on day j.

Let a_i be the number of days employee i is allowed to work in the scheduling period. Note that this constraint definition generalises the *number of days worked* constraint in problem $\mathcal{P}_1$, since now, different employees can be required to work a different number of days. Let d_{jk}^l, d_{jk}^u be the minimum, maximum number of employees required to work shift k on day j. The problem can be formulated as an integer linear program.

$$x_{ijk} = \begin{cases} 1 & \text{if employee } i \text{ works shift } k \text{ on day } j \\ 0 & \text{otherwise} \end{cases}$$

$$\mathcal{P}_2 : \min \sum_{i \in E} \sum_{j \in T} \sum_{k \in S} c_{ijk} x_{ijk} \tag{6}$$

$$\text{s.t.} \sum_{k \in S} x_{ijk} \leq 1 \quad \forall\, i \in E, j \in T \tag{7}$$

$$d_{jk}^l \leq \sum_{i \in E} x_{ijk} \leq d_{jk}^u \quad \forall\, j \in T, k \in S \tag{8}$$

$$\sum_{j \in T} \sum_{k \in S} x_{ijk} = a_i \quad \forall\, i \in E \tag{9}$$

$$\sum_{k \in S \backslash \bar{S}_{ij}} x_{ijk} = 0 \quad \forall\, i \in E, j \in T \tag{10}$$

$$x_{ijk} \in \{0, 1\} \quad \forall\, i \in E, j \in T, k \in S \tag{11}$$

The objective function (6) minimises the assignment costs. Constraints (7) limit the number of shifts assigned per employee and per

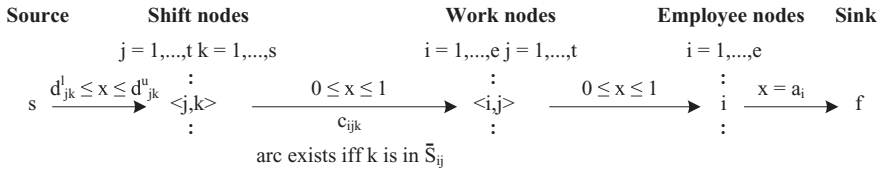


Fig. 3. Flow network G_2 for problem $\mathcal{P}_2$, x denotes the flow through an arc.

day to at most one. Constraints (8) are the coverage requirements. Constraints (9) restrict the number of days worked. Constraints (10) are the domain constraints. Constraints (11) require the decision variables to be either zero or one.

Problem $\mathcal{P}_2$ can be reformulated as a minimum cost network flow problem in a directed network $G_2 = (V, A)$, with V the set of nodes and A the set of arcs. The set V consists of five types of nodes: *shift nodes*, *work nodes*, *employee nodes*, one source node (s), and one sink node (f).

Fig. 3 shows the structure of G_2. Note that the domain constraints are modelled by only adding arcs between shift nodes associated with day j and shift k, and work nodes associated with employee i and day j, if $k \in \bar{S}_{ij}$. All nodes, except the source and sink nodes, are transshipment nodes. The supply in the source node is $\Sigma_{i \in E} a_i$, the supply in the sink node is equal to $\sum_{i \in E} -a_i$.

The total number of nodes in G_2 is $t(s + e) + e + 2$; the total number of arcs is $t(s + e(s + 1)) + e$, if $\bar{S}_{ij} = S$ for each $i \in E, j \in T$. Smaller sets $\bar{S}_{ij}$ will result in fewer arcs.

Lemma 2. *The size of network G_2 is polynomially bounded by the number of days, employees and shifts.*

Due to the configuration of G_2, a minimum cost solution respecting the capacity and demand constraints is equivalent to a solution for problem $\mathcal{P}_2$. Lower and upper bounds on the arc capacities are appropriately defined to correctly represent the minimum (maximum) staffing demand and the required number of days worked. Shift assignments correspond to flows in the arcs between the shift nodes and the work nodes. A flow from the shift node associated with day j, shift k to the work node associated with day j, employee i, corresponds to employee i working shift k on day j, thereby incurring cost c_{ijk}. As these arcs have a capacity upper bound of one, employees can be assigned to at most one shift per day.

Lemma 3. *A minimum cost flow in the network G_2 corresponds to an optimal solution for problem $\mathcal{P}_2$.*

From Lemmas 2 and 3, it follows that problem $\mathcal{P}_2$ can be transformed in polynomial time to a minimum cost network flow problem with integer capacities and arc costs. The resulting network flow problem can be solved in polynomial time (Ahuja et al., 1993), thereby establishing the following theorem.

Theorem 3. *Problem $\mathcal{P}_2$ can be solved in polynomial time.*

In the remainder of this section, two results will be derived from Theorem 3. First, a discussion on the complexity of academic nurse rostering benchmark datasets is presented. Second, a problem from the literature is revisited and a new network flow-based solution approach is presented.

3.2.1. Complexity of nurse rostering benchmark instances

Benchmark datasets provide interesting indicators for comparing the performance of different algorithms. Table 3 shows the hard constraints in three commonly used datasets in nurse rostering: the Nottingham dataset (NOTT; Brucker, Burke, Curtois, Qu, and Vanden Berghe, 2010), the dataset from the first International Nurse Rostering Competition (INRC; Haspeslagh, De Causmaecker, Schaerf, and Stølevik, 2012), and the KAHO dataset (Smet et al., 2014). Based on the hard constraints, INRC can be considered to be a special case of NOTT. Indeed, any algorithm that constructs a feasible solution for NOTT can construct feasible solutions for INRC.

Table 3
Hard constraints in benchmark instances.

	INRC	NOTT	KAHO
C1	Single assignment per day	Single assignment per day	Single assignment per day
C2	Coverage requirements	Coverage requirements	Qualification requirements
C3		Qualification requirements	Fixed assignments
C4			Only defined assignments
C5			No overlapping assignments

The NOTT instances can be straightforwardly transformed to problem $\mathcal{P}_2$ by setting a_i to the (maximum) number of days each employee i is allowed to work. Note that, if there is no constraint on the number of days worked, a_i can be set to the total number of days in the scheduling period, resulting in a feasible, albeit possibly poor quality, solution. The qualification requirements can be modelled using the domain constraints. The coverage requirements in NOTT can be defined as a range, minimum, maximum or exact number. To model these different definitions, the parameters d^l_{jk} and d^u_{jk} should be modified appropriately. An algorithm that generates a feasible solution for problem $\mathcal{P}_2$ can thus produce feasible solutions for NOTT and INRC. The following corollary is an immediate consequence of Theorem 3.

Corollary 1. *A feasible solution for the instances from the NOTT and INRC datasets can be obtained in polynomial time.*

The objective in NOTT and INRC is to minimise the weighted sum of soft constraint violations. These soft constraints are various time-related constraints limiting the number of consecutive days worked, weekends worked, etc. The cost of a solution thus depends on the extent to which certain constraints are violated, which is quite different from the assignment cost minimised in problem $\mathcal{P}_2$.

State of the art optimisation algorithms for personnel rostering often use a greedy method to construct an initial solution. Burke, Curtois, Qu, and Vanden Berghe (2013), for example, use a randomised greedy constructive algorithm to generate initial solutions by assigning uncovered shifts to the employee who incurs the smallest penalty gain. A simple example illustrates how this greedy method can fail to find a feasible solution: consider an instance without any soft constraints and with two employees A and B. Employee A has qualifications *DH* and *RN*, while employee B only has qualification *RN*. The coverage constraints require one *RN*-shift and one *DH*-shift to be assigned. If the greedy algorithm of Burke et al. (2013) selects employee A to be assigned to the *RN*-shift, the *DH*-shift cannot be assigned, and thus no feasible solution can be constructed without restarting the algorithm. However, by applying the method presented in Section 3.2, Corollary 1 states that a feasible solution can be guaranteed in polynomial time.

Corollary 1 cannot be extended to the KAHO dataset due to the hard constraint forbidding overlapping assignments (C5). The structure of network G_2 cannot prevent next-day overlap. All other hard constraints can be included in G_2 by making small modifications as

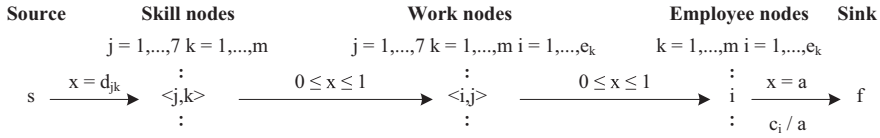


Fig. 4. Flow network $G^\star$ for a days-off scheduling problem with hierarchical substitution. x denotes the flow through an arc.

follows. Assignments can be fixed by changing the capacity bounds on the arcs between shift nodes and work nodes. Constraint C4 states that only assignments with a corresponding coverage requirement are feasible. This can be modelled in G_2 by adding shift nodes only if at least one employee is required on that day and shift.

3.2.2. A pseudo-polynomial time algorithm for a days-off rostering problem with hierarchical substitution

Hung (1994) and Billionnet (1999) describe a days-off rostering problem with hierarchical substitution. Based on their qualifications, employees are classified into m types, with e_k the number of employees of type k. Coverage requirements are defined in terms of these qualifications: d_{jk} is the number of workers required on day j with qualification k. The qualifications are organised in a hierarchical manner, i.e. a higher qualified employee can substitute for a lower qualified employee, but not the other way around. A cost c_k is associated with each type k employee. When an employee substitutes for a lower qualification, cost c_k still corresponds to the employee's original, higher qualification. The scheduling period is one week. Each employee should receive n days off, or, equivalently, each employee should work $a = 7 - n$ days. The objective is to find the least cost workforce composition, and to construct a days-off roster.

Hung (1994) presents sufficient conditions for a workforce composition to be feasible. A two phase approach is used for solving the problem: the workforce composition is determined first, and the actual roster is constructed afterward. An exhaustive search determines the workforce composition, suitable for problems with $m \leq 3$. Furthermore, a single pass method is described, which does not guarantee feasible solutions. Billionnet (1999) presents an integer programming formulation to determine the number of employees working a particular qualification each day. A feasible roster is constructed based on the solution for the integer program.

Given the least cost workforce, a roster can be efficiently constructed by reformulating and solving the problem as a minimum cost flow problem in network $G^\star$. Fig. 4 shows that $G^\star$ has a structure similar to G_2, however, *skills* are used instead of *shifts*. Arcs between skill nodes and work nodes are only present if the employee is qualified or can substitute for the skill. Lemma 3 holds for $G^\star$; a minimum cost flow solution in network $G^\star$ thus corresponds with a feasible roster.

Note that network $G^\star$ has a pseudo-polynomial number of nodes with respect to the coverage requirements of an instance, since the coverage requirements directly impact the workforce composition, and thus the number of work nodes and employee nodes.

Arcs between the employee nodes and the sink model the cost of each employee. These arcs have $l_{ij} = u_{ij} = a$ and a flow cost of c_i/a, with $c_i = c_k$ if employee i is of type k. If the employee is working, a units of flow result in a cost of $c_i/a \times a = c_i$, which is the employee's cost in the original problem definition.

3.3. Multiple shifts, varying demand (range), number of shifts worked of each type (range), domain constraints, feasibility

The constraint on the number of days worked discussed in Section 3.1 is a special case of *the number of shifts worked of each type* constraint. The former limits the number of assignments in the scheduling period, whereas the latter restricts the number of assignments of each shift type within the scheduling period. This constraint has various applications in practice, e.g. balancing undesirable shifts among employees or applying health and safety regulations.

Osogami and Imai (2000) discuss a feasibility problem with one counter constraint on the number of shifts worked of each type. This constraint is defined as a range, e.g. employee i has to be assigned to shift j on at least two days, and at most five days in the scheduling period. Furthermore, there are coverage requirements for each day, shift, also expressed as a range.

Let a^l_{ik}, a^u_{ik} be the minimum, maximum number of days employee i is allowed to work shift k. The problem can be formulated as an integer linear program.

$$x_{ijk} = \begin{cases} 1 & \text{if employee } i \text{ works shift } k \text{ on day } j \\ 0 & \text{otherwise} \end{cases}$$

$$\mathcal{P}_3 : \sum_{k \in S} x_{ijk} \leq 1 \quad \forall\, i \in E, j \in T \tag{12}$$

$$d^l_{jk} \leq \sum_{i \in E} x_{ijk} \leq d^u_{jk} \quad \forall\, j \in T, k \in S \tag{13}$$

$$a^l_{ik} \leq \sum_{j \in T} x_{ijk} \leq a^u_{ik} \quad \forall\, i \in E, k \in S \tag{14}$$

$$\sum_{k \in S \setminus \bar{S}_{ij}} x_{ijk} = 0 \quad \forall\, i \in E, j \in T \tag{15}$$

$$x_{ijk} \in \{0, 1\} \quad \forall\, i \in E, j \in T, k \in S \tag{16}$$

Constraints (12) ensure that at most one shift is assigned per day, per employee. Constraints (13) are the coverage requirements. Constraints (14) limit the number of shifts worked of each type. Constraints (15) are the domain constraints. Constraints (16) require the decision variables to be either zero or one.

Problem $\mathcal{P}_3$ is proven to be NP-complete by reduction from a timetabling problem (Osogami & Imai, 2000). However, by relaxing the constraint on the number of shifts worked of each type to a constraint on the number of days worked, problem $\mathcal{P}_3$ can be reduced to a feasible circulation problem (Ahuja et al., 1993) in a network obtained by modifying network G_2 in the following way. One additional arc is added to G_2 from the sink node to the source node with infinite positive capacity. Furthermore, the capacities on the arcs between employee nodes and sink node are modified to correctly model the range on the number of days worked.

The following theorem follows from Lemmas 2 and 3.

Theorem 4. *Problem $\mathcal{P}_3$ without ranged constraints on the number of shifts worked of each type can be solved in polynomial time.*

From Theorem 4 follows that the granularity of a counter constraint has a significant impact on a problem's complexity. Defining a counter for days worked allows the problem to be solved as a minimum cost network flow problem, whereas restricting the number of shifts worked of each type makes the problem NP-complete. This results in interesting practical considerations when solving personnel rostering problems. It might be sufficient to model regulations at day-level, rather than at shift-level, thereby inevitably making abstraction of some of the administration's guidelines. The result, however, is a computationally tractable problem.

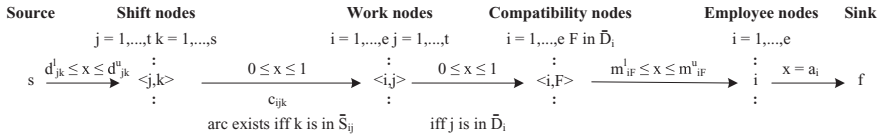


Fig. 5. Network G_4 corresponding with problem $\mathcal{P}_4$. x denotes the flow through an arc. Arcs from work nodes to employee nodes for days not in a set F are not drawn.

4. Succession constraints

This section presents results for a problem with succession constraints on days and shifts. First, a problem is discussed with a constraint that generalises the *number of days worked* constraint, and which can be used to model constraints on day successions. Afterward, the relationship between this problem and a known academic result is discussed. Van den Bergh et al. (2013) list 73 recent academic studies where constraints on day successions, shift successions and more general forbidden shift sequences appear.

As in Section 3, the titles of the subsections describe the problems discussed.

4.1. Multiple shifts, varying demand (range), number of days worked (exact), domain constraints, incompatible days (range), optimise preferences

Problem $\mathcal{P}_2$ can be extended by adding constraint (17) to restrict assignments on days from pairwise disjoint sets. This extended problem is denoted as $\mathcal{P}_4$.

$$m^l_{iF} \leq \sum_{j \in F} \sum_{k \in S} x_{ijk} \leq m^u_{iF} \quad \forall\, i \in E, F \in \bar{D}_i \tag{17}$$

For each employee i, let $\bar{D}_i$ be a set of day sets F, from which at least m^l_{iF}, and at most m^u_{iF} days can be worked. All sets in $\bar{D}_i$ must be pairwise disjoint, i.e. each day in T can occur in at most one set $F \in \bar{D}_i$. There should thus be no overlap between the sets in $\bar{D}_i$. The *incompatible days* constraint restricts the number of days worked by employee i in the set F to values between m^l_{iF} and m^u_{iF}.

This constraint can be interpreted as a *number of days worked* constraint for periods which are subsets of the scheduling period. This first interpretation allows various practical restrictions to be modelled, e.g. balancing the number of assignments per week, or limiting the number of Sundays worked in the scheduling period.

The incompatible day sets furthermore offer the opportunity to model succession constraints. Consider the example in which a hospital's administration requires a nurse to have a day-off after working on a bank holiday. For each bank holiday, a set F is added to $\bar{D}_i$ consisting of the bank holiday and the day after. By setting $m^l_{iF} = 0$ and $m^u_{iF} = 1$, nurse i will not be allowed to work on both the bank holiday and the day after. Note that, in this example, it is assumed that there are no consecutive bank holidays, since this would lead to non-disjoint sets in $\bar{D}_i$.

Problem $\mathcal{P}_4$ can be transformed into finding a minimum cost flow in a network G_4. This network is obtained by including additional *compatibility* nodes in network G_2. For each $F \in \bar{D}_i$, one compatibility node is added to the network between the work nodes for days in F, and the employee node of employee i. Since $\bar{D}_i$ is restricted to pairwise disjoint sets, the maximum size of $\bar{D}_i$ is bounded by t. The flow in the incoming arcs of each compatibility node is bounded between zero and one. The flow in the outgoing arc of each compatibility node is bounded between m^l_{iF} and m^u_{iF}. Fig. 5 shows the structure of the network G_4.

Theorem 5. *Problem $\mathcal{P}_4$ can be solved in polynomial time if $\bar{D}_i$ contains only pairwise disjoint sets of days.*

Proof. A large part of network G_4 is identical to G_2; the main ideas from Lemma 3 thus hold for G_4 as well. Since the size of $\bar{D}_i$ is bounded by t, the number of nodes in the network increases maximally with et. Through the construction of the compatibility nodes, each employee will be working between m^l_{iF} and m^u_{iF} days from the set F. □

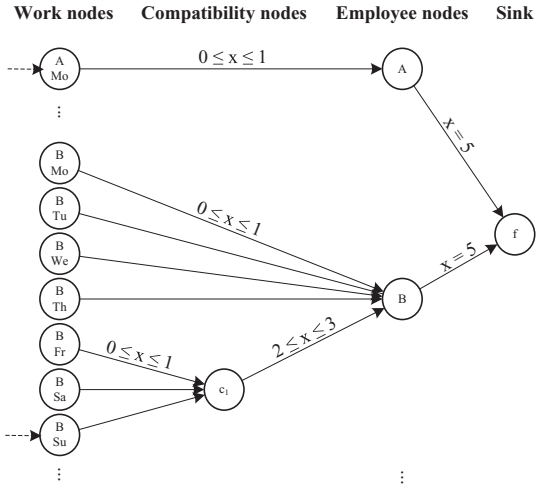


Fig. 6. Example of network G_4 in which employee B has to work at least two days in the period from Friday (Fr) to Sunday (Su).

Fig. 6 illustrates this formulation with an example. Each employee in the example has to work exactly five days, and employee B is required to work at least two days in the period from Friday (Fr) to Sunday (Su). To accommodate for this constraint, the compatibility node c_1 is included in the network.

The requirement for the sets to be pairwise disjoint is a strong modelling restriction. If non-disjoint sets would be allowed, additional constraints could be modelled with the presented networks. Consider the formulation of a *maximum m consecutive days worked* constraint in Equation (18).

$$\sum_{g \in \{0,\ldots,m\}} \sum_{k \in S} x_{i(j+g)k} \leq m \quad \forall\, i \in E,\ j \in \{1, \ldots, t-m\} \tag{18}$$

Equation (18) forces an employee to work at most m days in a window of size $m+1$, which slides over the scheduling period. Effectively, this formulation transforms the constraint on consecutive days worked into multiple constraints on non-disjoint incompatible days. However, non-disjoint sets F would result in multiple outgoing arcs from each work node in G_4, which would allow multiple assignments to one employee on one day, which violates the *single assignment per day* constraint.

4.2. Multiple shifts, varying demand (exact), domain constraints, shift succession constraints, feasibility

Health and safety regulations concerning rest time are of major importance in many organisations. Within this class of guidelines, providing sufficient rest time between two consecutive working days is regarded as one of the most important constraints. Forward shift rotation is a common concept in practice, which requires an assignment to not start earlier than the assignment on the previous day. A generalisation of this concept is a *shift succession* constraint, which forbids two particular shifts to be assigned on consecutive days.

Table 4
Comparison of characteristics of polynomially solvable problems from Lau (1996a; 1996b).

	Domain constraints	Unconstrained days-off roster	Shift succession	Day succession	Number of days worked
Lau (1996a)			✓		
Lau (1996b)		✓	(✓)[1]		
Theorem 5	✓	✓		(✓)[2]	✓

[1] Only monotonic shift succession constraints.
[2] Only pairwise disjoint day succession constraints.

Lau (1996b) discusses a feasibility problem in which the day-off roster has been predetermined. The goal is to assign shifts to working employees. Let $\bar{T}_i$ be a set of days on which employee i cannot work, i.e. the days-off in the predetermined roster. Let R be the set of shift pairs (k, k'), which cannot be assigned on two consecutive days. Coverage requirements are defined for each day and shift. They are expressed as an exact value. The problem can be formulated as the following integer linear program.

$$x_{ijk} = \begin{cases} 1 & \text{if employee } i \text{ works shift } k \text{ on day } j \\ 0 & \text{otherwise} \end{cases}$$

$$\mathcal{P}_5 : \sum_{k \in S} x_{ijk} \leq 1 \quad \forall\, i \in E, j \in T \setminus \bar{T}_i \tag{19}$$

$$\sum_{i \in E} x_{ijk} = d_{jk} \quad \forall\, j \in T, k \in S \tag{20}$$

$$x_{ijk} + x_{i(j+1)k'} \leq 1 \quad \forall\, i \in E, j \in T \setminus \{t\}, (k, k') \in R \tag{21}$$

$$\sum_{k \in S} x_{ijk} = 0 \quad \forall\, i \in E, j \in \bar{T}_i \tag{22}$$

$$\sum_{k \in S \setminus \bar{S}_{ij}} x_{ijk} = 0 \quad \forall\, i \in E, j \in T \tag{23}$$

$$x_{ijk} \in \{0, 1\} \quad \forall\, i \in E, j \in T, k \in S \tag{24}$$

Constraints (19) ensure that at most one shift is assigned to an employee on a working day. Constraints (20) are the coverage requirements. Constraints (21) are the shift succession constraints. Constraints (22) make sure no shifts are assigned on days-off. Constraints (23) are the domain constraints. Constraints (24) require the decision variables to be either zero or one.

Problem $\mathcal{P}_5$ is proven to be NP-complete by reduction from 3SAT (Lau, 1996b). Additionally, Lau (1996b) presents a greedy algorithm which solves $\mathcal{P}_5$ under three assumptions: a feasible solution exists, domain constraints are not included, and only monotonic shift changes are allowed. The latter is not a strong restriction as it still allows for forward shift rotation to be enforced.

The following theorem is due to Lau (1996b).

Theorem 6. *Problem $\mathcal{P}_5$ without domain constraints and with monotonic shift succession constraints can be solved in polynomial time (Lau, 1996b).*

In a follow-up paper, Lau (1996a) presents a polynomial time algorithm for problem $\mathcal{P}_5$ when the days-off roster has a particular structure. All employees' working days are contiguous and the work stretches either start or stop on the same day, i.e. the days-off roster is *tableau shaped*. A solution for this problem variant can be derived from an optimal path cover in a layered network.

The following theorem is due to Lau (1996a).

Theorem 7. *Problem $\mathcal{P}_5$ without domain constraints and with a tableau shaped days-off roster can be solved in polynomial time (Lau, 1996a).*

The network flow model presented in Section 3.2 supports the proof of a related result. By omitting the *shift succession* constraints, problem $\mathcal{P}_5$ can be reduced to the problem of finding a feasible flow in G_2. The fixed days-off roster can be modelled in the network by setting the capacity upper bounds of the correct arcs between work nodes and employee nodes to zero. Since the number of days worked per employee is predetermined in the days-off roster, a_i is set to $t - |\bar{T}_i|$, for each employee $i \in E$. Finally, the supply in the source node is set to $\Sigma_{j \in T} \Sigma_{k \in S} d_{jk}$. The supply in the sink node is equal to $\sum_{j \in T} \sum_{k \in S} -d_{jk}$. Note that neither the structure of the days-off roster, nor the domain constraints are subject to restrictions.

The following corollary is an immediate consequence of Lemmas 2 and 3.

Corollary 2. *Problem $\mathcal{P}_5$ without shift succession constraints can be solved in polynomial time.*

Table 4 compares characteristics of the polynomially solvable problems identified by Lau (1996a; 1996b), and by Theorem 5 presented in Section 4.1.

Relating the result from Lau (1996b) to Theorem 5 allows further examination of the boundary between easy and hard definitions of succession constraints. Problems with general *shift succession* constraints are NP-complete. However, Theorem 5 proved that a restricted version of the *day succession* constraints can be solved as a minimum cost flow problem. Similar to the findings on counter constraints, the transition from succession constraints on days to succession constraints on shifts transforms a tractable problem into an intractable one. While for counter constraints, in some cases, the abstraction from shifts to days could be justified (e.g. by aggregating constraints on the number of shifts worked of each type to restrict the total number of days worked), it is harder to do so for the succession constraints. Constraints on day successions are not useless, rather they have a different purpose than the shift successions. Nevertheless, this result provides valuable insight into the constraints that make personnel rostering problems hard.

5. Series constraints

Brunner et al. (2013) consider a problem with a single shift and varying demand, expressed as a minimum. In addition, the assignments of each employee are subject to three other constraints. The first constraint limits the number of days worked to a maximum value. The second and third constraints define a range on the number of consecutive days worked and days-off, respectively. The objective is to minimise the size of the workforce.

Van den Bergh et al. (2013) identify 101 recent academic studies that include constraints on the number of consecutive days worked or days-off.

Let $\bar{D}^{\text{work}}$ be the maximum number of days an employee can work in the scheduling period. Let $\bar{D}^{\text{on}}$, $\bar{D}^{\text{off}}$ be the maximum number of consecutive days worked, days-off. Let $\underline{D}^{\text{on}}$, $\underline{D}^{\text{off}}$ be the minimum number of consecutive days worked, days-off. The problem can be formulated as the following integer linear program.

$$x_{ij} = \begin{cases} 1 & \text{if employee } i \text{ works on day } j \\ 0 & \text{otherwise} \end{cases}$$

$$\mathcal{P}_6: \text{ min } \quad \text{number of employees} \tag{25}$$

$$\text{s.t.} \sum_{i \in E} x_{ij} \geq d_j \quad \forall j \in T \tag{26}$$

$$\sum_{j \in T} x_{ij} \leq \bar{D}^{\text{work}} \quad \forall i \in E \tag{27}$$

$$\sum_{n=0}^{\bar{D}^{\text{on}}} x_{i(j+n)} \leq \bar{D}^{\text{on}} \quad \forall i \in E, j \in \{1, \ldots, t - \bar{D}^{\text{on}}\} \tag{28}$$

$$x_{ij} + (w - \sum_{n=j+1}^{j+w} x_{in}) + x_{i(j+w+1)} \geq 1 \quad \forall i \in E, w \in \{1, \ldots, \underline{D}^{\text{on}} - 1\}, j \in \{1, \ldots, t - (w+1)\} \tag{29}$$

$$\sum_{n=0}^{\bar{D}^{\text{off}}} (1 - x_{i(j+n)}) \leq \bar{D}^{\text{off}} \quad \forall i \in E, j \in \{1, \ldots, t - \bar{D}^{\text{off}}\} \tag{30}$$

$$(1 - x_{ij}) + \sum_{n=j+1}^{j+w} x_{in} + (1 - x_{i(j+w+1)}) \geq 1 \quad \forall i \in E, w \in \{1, \ldots, \underline{D}^{\text{off}} - 1\}, j \in \{1, \ldots, t - (w+1)\} \tag{31}$$

$$x_{ij} \in \{0, 1\} \quad \forall\, i \in E, j \in T \tag{32}$$

The objective function (25) minimises the number of employees. Constraints (26) are the coverage requirements. Constraints (27) limit the number of days worked. Constraints (28) and (29) are the minimum and maximum number of consecutive days worked, respectively. Constraints (30) and (31) are the minimum and maximum number of consecutive days-off, respectively. Constraints (32) require the decision variables to be either zero or one.

Brunner et al. (2013) proved that problem $\mathcal{P}_6$ is NP-complete by showing that it has the *circulant problem* as a special case. Removing all constraints regarding consecutive assignments reduces the problem of finding a feasible solution for $\mathcal{P}_6$ to the problem of finding a feasible flow in network G_1. The lower bound on the flow in the arcs from employee nodes to the sink node needs to be set to zero, the upper bound should be set to $\bar{D}^{\text{work}}$.

The following corollary is an immediate consequence of Lemmas 1 and 2.

Corollary 3. *A feasible solution for problem $\mathcal{P}_6$ without ranged constraints on the number of consecutive days worked and days-off can be found in polynomial time.*

Limiting the number of consecutive days worked and days-off thus makes the problem hard. Consequently, almost all problems in practice are hard.

As was discussed in Section 4.1, the pairwise disjoint incompatible days constraint strongly resembles the constraint on the maximum number of consecutive working days. To strengthen Corollary 3, the same obstacle holds as was discussed in Section 4.1. Satisfaction of the *maximum number of consecutive working days* constraint cannot be guaranteed without the possibility of modelling a constraint on non-disjoint incompatible days in network G_4.

6. Conclusions and future research

The present paper systematically studied the complexity of personnel rostering problems, thereby further establishing the foundations for theoretical studies on models for rostering. By presenting transformations of different problems to minimum cost network flow problems, new cases were identified that can be solved in polynomial time. Specifically, decision and optimisation problems were reformulated with multiple shifts, varying demand, and constraints on the number of days worked, employees' domains and incompatible days.

Previously published complexity proofs were discussed in the light of these new results, and, as a result, a boundary between tractable and intractable personnel rostering problems was established. The new results show that for both counter constraints and succession constraints, the difference between easy and hard problems corresponds to defining constraints on day-level or on shift-level. These insights not only allow decision makers to reconsider the formulation of their problem, as it could mean making the problem computationally tractable, but also provide efficient approaches toward solving subproblems arising from decomposition.

The new contributions could be relevant outside personnel rostering as well, since the studied constraints also appear in other settings, mainly involving assignments of some type. In other personnel scheduling problems, tasks are assigned to workers, often subject to constraints on task changes (Ernst, Jiang, Krishnamoorthy, & Sier, 2004). In high school timetabling, the workload of a resource, e.g. student, teacher or room, is typically restricted by a minimum and maximum value (Post et al., 2012).

Future research should turn its attention toward problems with generalised constraints, e.g. restrictions on consecutive assignments, weekends. Models with such intricate constraints, can possibly no longer be transformed to the presented minimum cost flow problems.

References

Ahuja, R. K., Magnanti, T. L., & Orlin, J. B. (1993). *Network flows: Theory, algorithms, and applications*. Prentice Hall.

Bard, J. F., & Purnomo, H. W. (2005). Preference scheduling for nurses using column generation. *European Journal of Operational Research, 164*(2), 510–534.

Billionnet, A. (1999). Integer programming to schedule a hierarchical workforce with variable demands. *European Journal of Operational Research, 114*(1), 105–114.

Brucker, P., Burke, E. K., Curtois, T., Qu, R., & Vanden Berghe, G. (2010). A shift sequence based approach for nurse scheduling and a new benchmark dataset. *Journal of Heuristics, 16*(4), 559–573.

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

Brunner, J. O., Bard, J. F., & Köhler, J. M. (2013). Bounded flexibility in days-on and days-off scheduling. *Naval Research Logistics, 60*(8), 678–701.

Burke, E. K., Curtois, T., Qu, R., & Vanden Berghe, G. (2013). A time predefined variable depth search for nurse rostering. *INFORMS Journal on Computing, 25*(3), 411–419.

Burns, R. N., & Carter, M. W. (1985). Work force size and single shift schedules with variable demands. *Management Science, 31*(5), 599–607.

Ernst, A. T., Jiang, H., Krishnamoorthy, M., & Sier, D. (2004). Staff scheduling and rostering: A review of applications, methods and models. *European Journal of Operational Research, 153*(1), 3–27.

Haspeslagh, S., De Causmaecker, P., Schaerf, A., & Stølevik, M. (2012). The first international nurse rostering competition 2010. *Annals of Operations Research, 218*(1), 221–236.

Hung, R. (1994). Single-shift off-day scheduling of a hierarchical workforce with variable demands. *European Journal of Operational Research, 78*(1), 49–57.

Lau, H. C. (1996a). Combinatorial approaches for hard problems in manpower scheduling. *Journal of the Operations Research Society of Japan, 39*(1), 88–98.

Lau, H. C. (1996b). On the complexity of manpower shift scheduling. *Computers & Operations Research, 23*(1), 93–102.

Osogami, T., & Imai, H. (2000). Classification of various neighborhood operations for the nurse scheduling problem. In G. Goos, J. Hartmanis, J. Leeuwen, D. Lee, & S.-H. Teng (Eds.), *Lecture Notes in Computer Science: vol. 1969. Algorithms and computation* (pp. 72–83). Berlin Heidelberg: Springer.

Pentico, D. W. (2007). Assignment problems: A golden anniversary survey. *European Journal of Operational Research, 176*(2), 774–793.

Post, G., Ahmadi, S., Daskalaki, S., Kingston, J., Kyngas, J., Nurmi, C., et al. (2012). An XML format for benchmarks in high school timetabling. *Annals of Operations Research, 194*(1), 385–397.

Rocha, M., Oliveira, J., & Carravilla, M. (2013). Cyclic staff scheduling: optimization models for some real-life problems. *Journal of Scheduling, 16*(2), 231–242.
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.
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.
Vanhoucke, M., & Maenhout, B. (2007). NSPLib—A tool to evaluate (meta-) heuristic procedures. In S. Brailsford, & P. Harper (Eds.), *Operational research for health policy: Making better decisions, Proceedings of the 31st meeting of the European working group on operational research applied to health services* (pp. 151–165). Peter Lang.