Les Cahiers du GERAD ISSN: 0711–2440

Real-time personnel re-scheduling after a minor disruption

R. Hassani, G. Desaulniers,
I. Elhallaoui

G–2017–27

April 2017

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

Real-time personnel re-scheduling after a minor disruption

Rachid Hassani
Guy Desaulniers
Issmail Elhallaoui

GERAD & Department of Mathematics and Industrial Engineering, Polytechnique Montréal (Québec) Canada, H3C 3A7

rachid.hassani@gerad.ca
guy.desaulniers@gerad.ca
issmail.elhallaoui@gerad.ca

April 2017

Les Cahiers du GERAD
G–2017–27

Abstract: Personnel scheduling consists of determining least-cost employee work schedules to cover the demand of one or several jobs in each period of a time horizon. During the operations, several minor disruptions caused, for example, by a late employee may occur and must be addressed in real time by re-scheduling certain employees. In this paper, we develop a fast re-scheduling heuristic that can be used to solve the personnel re-scheduling problem in a context where the employees can be assigned to a wide variety of shifts such as in the retail industry. This heuristic considers five types of decision and is based on the dual values of the linear relaxation of the personnel scheduling problem. We also propose a procedure exploiting a multivariate adaptive regression splines method for updating the dual values after each disruption when several ones occur int he same week. Computational experiments conducted on a set of 1050 instances derived from real-life datasets involving up to 191 employees show the efficiency of the proposed re-scheduling heuristic: it can compute optimal solutions for more than 95% of these instances in less than one second on average. Furthermore, the dual value updating process allows an average reduction of 73% of the optimality gap for each disruption.

Acknowledgments: The authors are grateful to the personnel of Kronos Canadian Systems Inc. for proposing this problem and corresponding datasets. This work was funded by Kronos Canadian Systems and the Natural Sciences and Engineering Research Council of Canada. This financial support was greatly appreciated.

1 Introduction

Personnel scheduling problems are frequent in numerous domains such as retail, transportation, healthcare, and manufacturing, to name just a few. There exist several problem variants which are defined according to the environment in which they arise. For instance, in hospitals and factories, employees often work 24 hours per day, seven days per week and are assigned to a limited number of shifts (e.g., 7am to 3pm, 3pm to 11pm, and 11pm to 7am). In this case, the demand in employees for each job (task or position) to accomplish is expressed by shifts. On the other hand, in large retail stores or leisure resorts, demand occurs only during the opening hours and may highly fluctuate throughout a day and from one day to another. It needs to be expressed by short time intervals (e.g., 15-minute periods) for each job. To meet these demand fluctuations, numerous shifts, starting and ending at about any time, can be assigned to the employees. Also, in certain contexts, an employee is always assigned to the same job in a shift while, in others, he/she can be assigned to multiple jobs in a shift. Typically, a personnel scheduling problem consists of determining the working schedules (the exact shifts and jobs) of a set of employees such that the demand for each job and each period is met as much as possible at minimum costs while satisfying labor rules. Costs include employee salaries and often various penalties for enforcing the satisfaction of soft constraints or preferences. Solving these problems is often a challenge that is addressed using an optimization algorithm.

Unfortunately, a personnel schedule is rarely operated as planned. Indeed, it often occurs that an employee is late for work, must leave earlier than scheduled, or simply does not show up. Such a *minor disruption* usually makes the planned schedule infeasible and must be addressed as soon as possible. To deal with a minor disruption, re-scheduling must be performed, i.e., the planned schedule must be modified to retrieve a feasible schedule that takes into account the disruption but does not differ much from the planned schedule. Given that minor disruptions are often known just before the operations and the modified employee schedules must be announced to the impacted employees as soon as possible, re-optimization must be done in real time.

In this paper, we address the problem of re-optimizing a personnel schedule to address a minor disruption in real time. We assume that this problem occurs in a context where there are several jobs to cover by skilled employees, the number of employees required for each job fluctuates throughout the planning horizon, and the employees can be assigned to a wide variety of possible shifts, starting and ending at various times. In this case, the disruption can cause a lack of personnel for a job just for a few periods (say, one hour) and can thus be addressed, for example, by simply extending the shift of an employee to cover these periods. Other options can be considered to resolve a minor disruption but, in all cases, the number of changes brought to the planned schedules must be rather limited.

1.1 Literature review

The study of personnel scheduling problems in the operations research literature started at the beginning of the 1950s with the work of Edie [6] who introduced a heuristic method for solving a shift scheduling problem for toll booth operators with the objective of minimizing waiting at the toll booths. A few months later, Dantzig [5] proposed to model this problem as a set covering problem and solve it using integer linear programming techniques. Since then, numerous papers have been published on many personnel scheduling problems. Exhaustive surveys on these works were written by Ernst et al. [7], Burke et al. [2], and Van den Bergh et al. [16].

Literature on personnel re-scheduling is rather scarce. Most works have tackled the nurse re-rostering problem that considers a very limited number of possible shifts (three shifts per day in most cases) and, often, demand by shifts. In this case, a minor disruption corresponds to the absence of a nurse for one or several consecutive shifts and can be resolved by re-assigning the nurses to the shifts for one or several days, or resorting to part-time or on-call nurses. Exact branch-and-bound and branch-and-price algorithms for solving variants of this problem were proposed by Moz and Pato [12, 13] and Bard and Purnomo [1]. These algorithms require too large computational times to be applicable in real time. Heuristic algorithms were also developed by Moz and Pato [12, 14], Pato and Moz [15], Maenhout and Vanhoucke [11], and Kitada and

Morizawa [10]. Even if some of these algorithms yield fast computational times, they are suited for the nurse re-rostering context which differs from the context addressed in this paper.

Personnel re-scheduling has also been studied in other sectors such as transportation (see Clausen et al. [4] and Cacchiani et al. [3] for surveys in air and rail transportation, respectively). Because the crews travel during their working days and may even overnight outside of their homes in some cases, these problems differ substantially from the one considered in this study and, therefore, the proposed models and solution approaches are not suitable.

To the best of our knowledge, no works have been published on personnel re-scheduling problems arising in retail stores or similar settings, beside the recent master's dissertation of Froger [9]. This dissertation focuses on large disruptions, such as when a relatively large variation of the demand is expected to occur on one or several days due, for instance, to bad weather or an unanticipated sale. The author developed an integer linear program that considers variables associated with the planned employee shifts and additional shifts that are derived from the planned shifts according to parameterized modification rules. Different rules and parameter values are tested. In all cases, the resulting model can be solved by a commercial mixed integer programming solver in less than two minutes for instances involving up to 50 employees. This work has similarities with our study. The initial personnel scheduling problem is the same and the modifications that can be brought to the planned shifts are the same. On the other hand, the sizes of the disruptions differ and the time available to address a disruption is much less in our case (a few seconds versus a few minutes). Consequently, the solution approach proposed by Froger [9] cannot be used for tackling a minor disruption in real time.

1.2 Contributions

The previous literature review shows that personnel re-scheduling following a minor disruption in a retail store or a similar environment (where demand is highly fluctuating, numerous shifts are feasible, planned shifts can be modified in various ways, and fast computational times are required) has not been addressed yet. This paper aims at fulfilling this gap. First, we develop an efficient heuristic for re-optimizing a personnel schedule in real time once a minor disruption becomes known. This fast heuristic can compute for a single disruption one or several solutions. It is based on the dual solution of the linear relaxation of the model used to compute the planned schedule. Second, we exploit a non-parametric regression method for estimating the evolution of some dual values when the planned schedule is subject to multiple disruptions which yield a sequence of re-optimizations. Third, through extensive computational experiments performed on various instances derived from real-world datasets involving between 15 and 195 employees, we show that the proposed heuristic finds in less than two seconds an optimal solution in more than 95% of the test cases. Additional computational results obtained when resolving a sequence of minor disruptions occurring on the same planned schedule indicate that using the proposed regression method for updating the dual values reduces the gap between the optimal value and the heuristic solution value by an average of 73%.

1.3 Paper structure

This paper is organized as follows. In Section 2, we first state the initial personnel scheduling problem and then derive from it the personnel re-scheduling problem considered. In Section 3, we describe the proposed heuristic and the non-parametric regression method used to update dual values. In Section 4, we report and analyze the results of our computational experiments. Finally, conclusions are drawn in Section 5.

2 Problem description

In this section, we start by stating the personnel scheduling problem solved to establish planned schedules for the employees. We also model this problem as an integer program that will be used to justify the proposed heuristic. Finally, we describe the personnel re-scheduling problem by defining the minor disruptions considered and the type of decisions that can be made to resolve them.

2.1 The personnel scheduling problem

A few days, a week or, in some cases, a month before the planning horizon (typically, one week or one month), the personnel manager must disclose the employee working schedules for that horizon, i.e., the days that each employee will work and, for each working day, the exact shift that he/she will work and on which job. Let us present a detailed definition of this personnel scheduling problem when the horizon $\mathcal{H} = \{1, 2, ..., 7\}$ is a set of seven consecutive days numbered from 1 to 7. The model and methodology described below can easily be adapted for other horizons.

Horizon $\mathcal{H}$ is also divided into a set $\mathcal{P} = \{1, \ldots, p^{max}\}$ of consecutive periods of equal length (for example, 15 minutes) numbered from 1 to p^{max}. Any shift that can be assigned to an employee starts at the beginning of a period and ends at the end of another period. Let $\mathcal{E}$ be the set of available employees and $\mathcal{W}$ the set of jobs to accomplish. In the retail context, a job is a type of a position that serves the customers in real time such as a cashier in a supermarket or a clerk in the bakery department. An employee assigned to a job can be replaced at any time by another employee or can interrupt its assignment at any time to go for a break, move to another job, or end his/her day.

In this paper, we assume that an employee can be assigned to a single job during a shift and that no breaks need to be scheduled in the shifts. This first assumption is common in practice and is imposed only for ease of exposition as our heuristic can be easily adapted if it does not hold. The second assumption is also frequent in practice because the breaks are often assigned during the operations depending on the observed demand. Typically, to take breaks into account when planning the personnel schedule, the forecasted demand is increased during certain time intervals.

The main goal of the personnel scheduling problem is to assign the employees to shifts such that enough employees are on duty for each job in each period so as to offer a service of good quality to the customers. One of the main input to this problem is, thus, the required number of employees d_w^p for each job $w \in \mathcal{W}$ and period $p \in \mathcal{P}$. To fulfill this demand, one must determine for each employee $e \in \mathcal{E}$ a working schedule that is composed of shifts taken from sets of potential feasible shifts $\mathcal{S}_e^h$, $h \in \mathcal{H}$, that can be assigned to e. A shift s is defined by a starting period b_s, an end period f_s, a length l_s in number of periods ($l_s = f_s - b_s + 1$), and a job w_s to which the employee is assigned in this shift. It belongs to $\mathcal{S}_e^h$ if: 1) b_s belongs to day h and is an admissible starting period for employee e; 2) l_s falls into a feasible shift length interval (for instance, between 4 and 10 hours); and 3) employee e is skilled for job w_s and available on day h. Like the employee skills, the set of admissible starting periods and the feasible shift length interval may vary from one employee to another. The schedule of employee e is feasible if it contains: 1) at least n^O days off; 2) at most one shift per working day; 3) at least n^R periods of rest between shifts assigned on two consecutive days; and 4) a total of at most n^P working periods. Note that, for horizons longer than one week, additional conditions on the distribution of the days off need to be considered and the maximum number of periods may be imposed for every week, not for the whole horizon.

Given that the demand for each job may fluctuate from one period to the next, it is nearly impossible to cover this demand exactly with a set of feasible employee schedules. In fact, it is not rare that there is too much or too many employees assigned to a job in a period. In the former case, we say that each exceeding employee yields an over-covering. In the latter, each missing employee corresponds to an under-covering. In practice, the under-coverings are usually covered by temporary or on-call employees who are usually scheduled just before the operations if really needed. To exploit this possibility at best, the under-coverings for a job should be bunched together as much as possible so as to be able to cover by shifts for these extra employees. Consequently, the input to the personnel scheduling problem also includes a set of anonymous shifts, denoted S^A, that can be eventually assigned to these employees, i.e., which are feasible with respect to their start times and lengths. Beside a starting period b_s, an end period f_s, and a length l_s, an anonymous shift $s \in S^A$ is also defined by a job w_s. Considering these shifts allows to forbid demand under-covering. They should, however, be highly penalized. Note that an anonymous shift can be chosen several times in a solution.

The personnel scheduling problem involves three types of costs: labor costs, anonymous shift penalties, and over-covering penalties. The labor costs correspond to the salaries of the employees that are paid for the time worked. Typically, the most senior employees, those that should be favored to work more, have the largest hourly rates. Consequently, if the real hourly rates were considered in the model, the senior employees would be assigned less working hours than the junior ones. To avoid this, one can consider an hourly rate that is the same for all employees. In our case, to favor an almost even distribution of the working hours among the employees, we use a step-wise function to compute an adjusted labor cost for each employee, that is, the hourly rate increases by step with the number of hours worked. This function is defined by a set of steps $\mathcal{K}^L = \{1, \ldots, m^L\}$, where m^L is the number of steps, and, for each step k, a number of periods n_k^L which define the length of this step and a cost per period c_k^L such that $c_k^L < c_{k+1}^L$ for $k \in \mathcal{K}^L \setminus \{m^L\}$. Notice that $\sum_{k\in\mathcal{K}^L} n_k^L$ corresponds to the maximum number of periods that an employee can work in a week.

Next, each anonymous shift $s \in \mathcal{S}^A$ yields a penalty c_s^Y that is proportional to the shift length l_s. These penalties are large enough compared to the other costs to ensure the minimization of the total number of periods assigned in anonymous shifts. Finally, over-covering penalties are also considered even though the labor costs ensure that there will be no extra over-covering. In fact, we use again a step-wise function to compute the over-covering penalty associated with each job in each period. This function allows to distribute almost evenly the over-covering of a job among the various periods. For instance, if two periods of over-covering are unavoidable for a job, it is preferable to have them, if possible, on two different periods rather than at the same period. The over-covering penalty function does not depend on the job and is defined by a set of steps $\mathcal{K}^V = \{1, \ldots, m^V\}$, a number of over-coverings n_k^V in each step k, and a cost per over-covering c_k^V in each step k with $c_k^V < c_{k+1}^V$ for $k \in \mathcal{K}^V \setminus \{m^V\}$.

To model the personnel scheduling problem, we also rely on the following notation. For each period $p \in \mathcal{P}$, job $w \in \mathcal{W}$ and shift $s \in \mathcal{S}^A \cup \bigcup_{h\in\mathcal{H}, e\in\mathcal{E}} \mathcal{S}_e^h$, the binary parameter a_{sw}^p takes value 1 if shift s covers job w in period p and 0 otherwise. Let M be a sufficiently large constant. The variables are defined as follows.

- X_{es}^h: Binary variable equal to 1 if shift $s \in \mathcal{S}_e^h$ is assigned to employee $e \in \mathcal{E}$ on day $h \in \mathcal{H}$ and 0 otherwise;
- O_e^h: Binary variable equal to 1 if employee $e \in \mathcal{E}$ has a day off on day $h \in \mathcal{H}$ and 0 otherwise;
- L_e^k: Integer variable specifying the number of periods worked by employee $e \in \mathcal{E}$ and that fall on step $k \in \mathcal{K}^L$;
- Y_s: Integer variable indicating the number of times that anonymous shift $s \in \mathcal{S}^A$ is used in the solution;
- V_w^{kp}: Integer variable indicating the number of over-coverings for job $w \in \mathcal{W}$ in period $p \in \mathcal{P}$ that fall on step $k \in \mathcal{K}^V$.

Given this notation, the personnel scheduling problem can be modeled as the following integer program.

$$\text{Minimize} \quad \sum_{e\in\mathcal{E}} \sum_{k\in\mathcal{K}^L} c_k^L L_e^k + \sum_{s\in\mathcal{S}^A} c_s^Y l_s Y_s + \sum_{p\in\mathcal{P}} \sum_{w\in\mathcal{W}} \sum_{k\in\mathcal{K}^V} c_k^V V_w^{kp} \tag{1}$$

$$\text{subject to :} \quad \sum_{s\in\mathcal{S}_e^h} X_{es}^h + O_e^h = 1, \qquad \forall e \in \mathcal{E}, h \in \mathcal{H} \tag{2}$$

$$\sum_{h\in H} O_e^h \geq n^O, \qquad \forall e \in \mathcal{E} \tag{3}$$

$$\sum_{h\in\mathcal{H}} \sum_{s\in\mathcal{S}_e^h} l_s X_{es}^h - \sum_{k\in\mathcal{K}^L} L_e^k = 0, \qquad \forall e \in \mathcal{E} \tag{4}$$

$$M\, O_e^{h+1} + \sum_{s\in\mathcal{S}_e^{h+1}} b_s X_{es}^{h+1} - \sum_{s\in\mathcal{S}_e^h} (f_s + 1 + n^R) X_{es}^h \geq 0, \qquad \forall e \in \mathcal{E}, h \in \mathcal{H} \setminus \{7\} \tag{5}$$

$$\sum_{e\in\mathcal{E}} \sum_{h\in\mathcal{H}} \sum_{s\in\mathcal{S}_e^h} a_{sw}^p X_{es}^h + \sum_{s\in\mathcal{S}^A} a_{sw}^p Y_s - \sum_{k\in\mathcal{K}^V} V_w^{kp} = d_w^p, \qquad \forall p \in \mathcal{P}, w \in \mathcal{W} \tag{6}$$

$$X_{es}^h \in \{0,1\}, \qquad \forall e \in \mathcal{E}, h \in \mathcal{H}, s \in \mathcal{S}_e^h \tag{7}$$

$$O_e^h \in \{0,1\}, \qquad \forall e \in \mathcal{E}, h \in \mathcal{H} \tag{8}$$

$$L_e^k \in [0, n_k^L], \text{integer}, \qquad \forall e \in \mathcal{E}, k \in \mathcal{K}^L \tag{9}$$

$$Y_s \geq 0, \text{integer}, \qquad \forall s \in \mathcal{S}^A \tag{10}$$

$$V_w^{kp} \in [0, n_k^V], \text{integer}, \qquad \forall w \in \mathcal{W}, p \in \mathcal{P}, k \in \mathcal{K}^V. \tag{11}$$

The objective function (1) aims at minimizing the sum of the labor costs, the penalties incurred by the anonymous shifts, and the over-covering penalties. Constraints (2) ensure that each employee is assigned to a shift or a day off on each day of the horizon. Constraints (3) impose the assignment of at least n^O days off to each employee. The distribution of the periods work by each employee on the various steps of $\mathcal{K}^L$ are computed through constraints (4). Note that the relations between the costs c_k^L ensure that the cheapest steps are always used first. Constraints (5) specify that there must be a rest of at least n^R periods between shifts assigned to the same employee on two consecutive days. For each job and each period, constraints (5) guarantee that the demand is covered by a sufficient number of employees or anonymous shifts. They also allow to compute the number of over-coverings per step in $\mathcal{K}^V$. Finally, the domains of the decision variables are restricted by (6)–(11).

2.2 The personnel re-scheduling problem

In general, the small disruptions which can make a planned schedule infeasible are numerous. These disruptions can be triggered, for instance, by the lateness or the absence of an employee, a demand increase at certain periods of a day, or an unforeseen event that prevents an employee to complete a planned work shift. In this paper, we focus on the lateness of an employee because the other cases can be treated similarly. Indeed, in all cases, the personnel schedule must be adjusted to cover a demand for a certain number of consecutive periods.

In the following, we characterize a minor disruption due to a lateness by:

- the employee $\hat{e} \in \mathcal{E}$ who is late;
- the day $\hat{h} \in \mathcal{H}$ when the disruption occurs;
- the planned shift $\hat{s} \in \mathcal{S}_{\hat{e}}^{\hat{h}}$ of employee $\hat{e}$;
- the lateness duration $\hat{l}$ in periods ($\hat{l} < l_{\hat{s}}$);
- the period $\hat{p} \in \mathcal{P}$ at which the manager is notified that the employee will be late. We assume that $\hat{p}$ is a period prior to $b_{\hat{s}}$, possibly the period just preceding it.

Hereafter, we call the first $\hat{l}$ periods of shift $\hat{s}$ the *uncovered demand block*.

Each company has its own set $\mathcal{D}$ of possible types of decision that can be applied to update its personnel schedule following a minor disruption $(\hat{e}, \hat{h}, \hat{s}, \hat{l}, \hat{p})$. Each type $d \in \mathcal{D}$ is associated with a set of candidates $\mathcal{Q}_d(\hat{e}, \hat{h}, \hat{s}, \hat{l}, \hat{p})$. A candidate $q \in \mathcal{Q}_d(\hat{e}, \hat{h}, \hat{s}, \hat{l}, \hat{p})$ is a subset $\mathcal{E}^q$ of employees (possibly just one) excluding employee $\hat{e}$ whose schedules can be modified on day $\hat{h}$ to address the disruption without violating any feasibility constraint. Because the disruption is minor, the modifications must only impact a limited number of employees and only on day $\hat{h}$. Given a candidate, there might exist several combinations of modified shifts to assign to them in order to resolve the disruption. For ease of exposition, we assume that the best combination can easily be found. Each decision type $d \in \mathcal{D}$ incurs a managerial cost γ_d, which usually accounts for the time spent implementing the schedule changes. These costs can also be used to represent the preferences of the manager for certain decision types. The personnel re-scheduling problem consists of determining the candidate among all decision types that yields the smallest cost increase. This increase is computed as the difference between the cost of the modified schedule (including the managerial cost) and that of the planned schedule.

In this paper, we consider five basic decision types, i.e., $\mathcal{D} = \{d_1, \ldots, d_5\}$. The proposed methodology can, however, be adapted to other types. The five studied types are the following.

- d_1: Extend the planned shift $s \in \mathcal{S}_e^{\hat{h}}$ of an employee e working on day $\hat{h}$ to cover the uncovered demand block. Shift s must end at period $\hat{p}$ or after but before $b_{\hat{s}}$.
- d_2: Assign a shift $s \in \mathcal{S}_e^{\hat{h}}$ that covers the uncovered demand block to an employee e who has a planned day off on day $\hat{h}$. This is possible only if employee e has sufficient time between $\hat{p}$ and $b_{\hat{s}}$ to reach work from home.
- d_3: Switch shift $\hat{s}$ with the planned shift $s \in \mathcal{S}_e^{\hat{h}}$ of an employee e who is planned to start after the late arrival of employee $\hat{e}$, i.e., such that $b_s \geq b_{\hat{s}} + \hat{l}$. Here also, this is possible only if employee e has sufficient time between $\hat{p}$ and $b_{\hat{s}}$ to arrive at work.
- $d_4(\eta)$: Extend or move forward the planned shifts of at most η employees whose shifts are scheduled to end between $\hat{p}$ and $b_{\hat{s}}$. Figure 1) gives an example where moving forward the shift employee e_1 might be necessary to avoid reaching the maximum working time per week. Parameter η is set by the manager and limits the number of employees involved in this type of decision. Note that type d_4 generalizes type d_1 as $d_4(1) = d_1$.
- $d_5(\eta)$: Switch the planned shifts of at most $\eta + 1$ employees (including employee $\hat{e}$) such that employee $\hat{e}$ is re-assigned to a shift s with $b_s \geq b_{\hat{s}} + \hat{l}$ (see Figure 2). Here again, η imposes a limit on the number of employees involved in the decision (beside employee $\hat{e}$). This decision type generalizes type d_3 as $d_5(1) = d_3$.

3 Heuristic

In this section, we describe the proposed heuristic to solve the personnel re-scheduling problem. This heuristic is based on dual variable values obtained when solving the personnel scheduling problem. First, we consider the case where a single minor disruption needs to be addressed. Then, we discuss the case where a sequence of minor disruptions must be handled without re-optimizing the personnel scheduling problem (to obtain updated dual values) after each disruption.

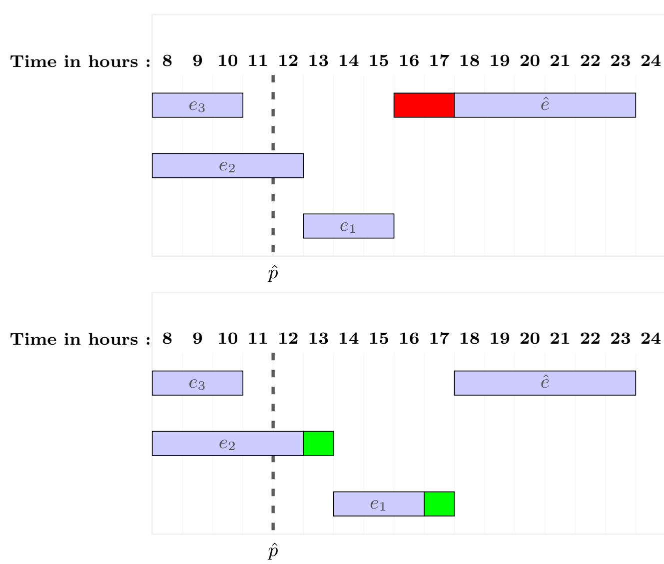

Figure 1: Example of a decision of type $d_4(2)$ for a candidate $q = \{e_1, e_2\}$. The top part shows the planned shifts with the uncovered demand block in red. In the bottom part, the shift of employee e_2 has been extended while that of employee e_1 has been moved forward and extended

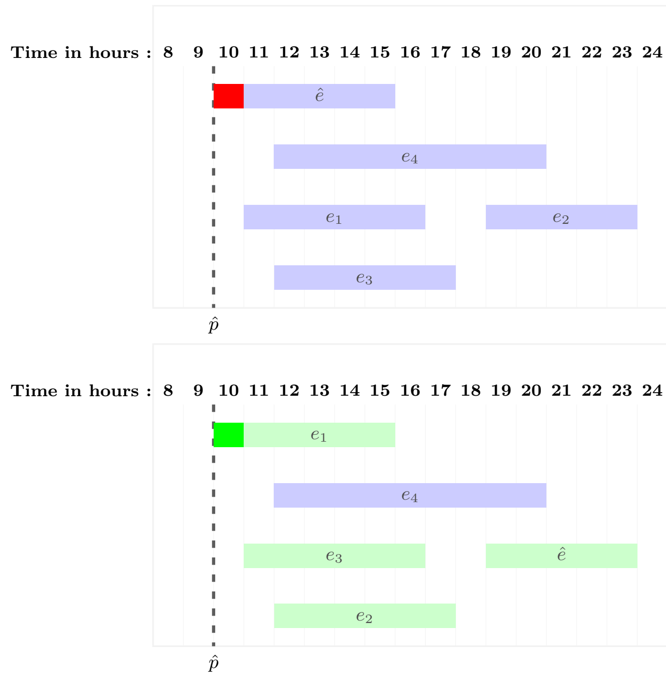

Figure 2: Example of a decision of type $d_5(3)$ for a candidate $q = \{e_1, e_2, e_3\}$. The top part shows the planned shifts with the uncovered demand block in red. The bottom part shows the employee shifts after re-scheduling

3.1 A single disruption

Let us consider a minor disruption defined by $(\hat{e}, \hat{h}, \hat{s}, \hat{l}, \hat{p})$. The proposed heuristic is based on an enumeration of all possible candidates for all applicable decision types. It is described in Algorithm 1. In this algorithm, d^* and q^* indicate the best decision type and the best candidate of this type, respectively, yielding an estimated cost increase of Δ^*. When building the set of candidates for a decision type in Step 4, we check for each candidate that the modified shift s of an employee e in this candidate belongs to $\mathcal{S}_e^{\hat{h}}$ and that constraints (3) and (5) remain satisfied. In Step 6, the computation of the cost increase estimate Δ_d^q for a candidate q depends on the decision type d as discussed below. Note that, for a given disruption, one can restrict in Step 3, the set of decision types that can be applied. For instance, if the uncovered demand block lasts only 30 minutes, then decision type d_2 might not be interesting given that a guaranteed minimum time largely exceeding 30 minutes might have to be paid. Furthermore, observe that the algorithm can easily be adapted to output not only the best found decision, but several decisions from which the manager can choose the one he/she prefers.

Algorithm 1 Re-scheduling heuristic

1: Given a minor disruption $(\hat{e}, \hat{h}, \hat{s}, \hat{l}, \hat{p})$
2: Set $\Delta^* := \infty$, $d^* := NIL$, $q^* := NIL$
3: **for** each decision type $d \in \mathcal{D}$ **do**
4:     Build the set of candidates $\mathcal{Q}_d(\hat{e}, \hat{h}, \hat{s}, \hat{l}, \hat{p})$
5:     **for** each candidate $q \in \mathcal{Q}_d(\hat{e}, \hat{h}, \hat{s}, \hat{l}, \hat{p})$ **do**
6:         Estimate the cost increase Δ_d^q
7:         **if** $\Delta_d^q < \Delta^*$ **then**
8:             $\Delta^* := \Delta_d^q$, $d^* := d$, $q^* := q$
9: Return (Δ^*, d^*, q^*)

The formulas used to compute the cost increase estimates Δ_d^q are derived from the following linear programming theory. Let us rewrite model (1)–(11) in the following compact form:

$$\begin{array}{lll} \text{(P)} & \min\limits_{x} & c^\top x \\ & \text{subject to:} & Ax = b \\ & & x \in \mathbb{N}^n, \end{array}$$

where x is the vector of all n decision variables, c is the vector of the corresponding cost coefficients, A is the constraint coefficient matrix, and b is the constraint right-hand side vector. Denote by R the linear relaxation of P. Let B be an optimal basis for R and write $A \equiv [B, N]$, where B and N are the submatrices associated with the basic and nonbasic variables, respectively. Vectors x and c are also partitioned accordingly, i.e., $x^\top \equiv [x_B^\top, x_N^\top]$ and $c^\top \equiv [c_B^\top, c_N^\top]$. Let be the vector of dual variables. Basis B yields the optimal solution x^* with $x_B^* = B^{-1}b$ and $x_N^* = \vec{0}$ and is such that $c_N^\top - (\pi_B^*)^\top N \geq (\vec{0})^\top$, where $(\pi_B^*)^\top = c_B^\top B^{-1}$ is the corresponding optimal dual solution. The cost of these primal and dual solutions is $z^* = c_B^\top x_B^* = (\pi_B^*)^\top b$.

From the linear programming sensitivity analysis theory, we know that basis B may remain optimal for a perturbed model obtained by slightly modifying the right-hand side vector b, i.e., by replacing b with $b + \delta b$, where δb is a perturbation vector. In this case, the modified primal solution is given by $\tilde{x}_B^* = B^{-1}(b + \delta b) = x_B^* + B^{-1}\delta b$ and its cost by $\tilde{z}^* = c_B^\top \tilde{x}_B^* = z^* + (\pi_B^*)^\top \delta b$. The cost increase yielded by perturbation vector δb is, thus, equal to $\tilde{z}^* - z^* = (\pi_B^*)^\top \delta b$.

For a candidate q of the decision type d, we compute the cost increase estimate as follows:

$$\Delta_d^q = \gamma_d + (\pi_B^*)^\top \delta b_d^q, \tag{12}$$

where the perturbation vector δb_d^q is defined according to the shift variables $X_{es}^{\hat{h}}$, $e \in \mathcal{E}^q \cup \{\hat{e}\}$, $s \in \mathcal{S}_e^{\hat{h}}$, switching either from one to zero or from zero to one in the decision associated with candidate q. More precisely, let A_{es}^h be the column of matrix A associated with a variable X_{es}^h. Furthermore, let $\mathcal{Z}_d^q$ and $\bar{\mathcal{Z}}_d^q$ be the sets of the indices (e, s) of the shift variable $X_{es}^{\hat{h}}$ switching from one to zero and from zero to one, respectively. Then,

$$\delta b_d^q = \sum_{(e,s) \in \bar{\mathcal{Z}}_d^q} A_{es}^{\hat{h}} - \sum_{(e,s) \in \mathcal{Z}_d^q} A_{es}^{\hat{h}}. \tag{13}$$

We believe that the estimate provided by (12) is good because we have observed that the integrality gap yielded by model (1)–(11) is very small (see Table 1) and, thus, x_B and the optimal solution of P are quite similar. Furthermore, given that the considered disruption is minor, the norm of δb is small and a large number of the columns forming the basis B should remain in the optimal basis of the perturbed model.

In the rest of this section, we present the explicit formula (12) for each decision type $d \in \mathcal{D}$. We denote by $\pi_{eh}^{(2)}$, $\pi_e^{(4)}$, $\pi_{eh}^{(5)}$, and $\pi_{pw}^{(5)}$ the computed optimal dual values associated with constraints (2), (4)–(5), respectively. We also define $\pi_{e0}^{(2)} = \pi_{e0}^{(5)} = 0$ for all $e \in \mathcal{E}$.

Decision type d_1: For a candidate q of this type, shift $\hat{s}$ for the late employee $\hat{e}$ is replaced by a shift obtained from $\hat{s}$ by omitting the uncovered demand block. Hence, employee $\hat{e}$ works $\hat{l}$ periods less and starts $\hat{l}$ periods later than planned. To cover the uncovered block, the end of the shift s of an employee e is extended by $t_e = \hat{l} + b_{\hat{s}} - f_s - 1$ periods, yielding an additional over-covering for job $w_{\hat{s}}$ in each period from $f_s + 1$ to $b_{\hat{s}} - 1$ if $b_{\hat{s}} - f_s \geq 2$. Consequently, the cost increase estimate is given by

$$\Delta_{d_1}^q = \gamma_{d_1} + \hat{l}\pi_{\hat{e}}^{(4)} - t_e\pi_e^{(4)} - \hat{l}\pi_{\hat{e},\hat{h}-1}^{(5)} + t_e\pi_{e,\hat{h}}^{(5)} + \sum_{p=f_s+1}^{b_{\hat{s}}-1} \pi_{pw_{\hat{s}}}^{(5)}.$$

Decision type d_2**:** For a candidate q of this type, shift $\hat{s}$ for the employee $\hat{e}$ is also replaced by a shorter shift that omits the uncovered demand block. The uncovered block is then covered by a shift s of an employee e that was assigned to a day off on day $\hat{h}$. This shift starts at period $b_{\hat{s}}$ and lasts $\hat{l}$ periods. In practice, such a decision type is admissible only if $\hat{l}$ is greater or equal to the minimum number of periods in a feasible shift. For this decision, the cost increase estimate expresses as:

$$\Delta^q_{d_2} = \gamma_{d_2} - \pi^{(2)}_{e\hat{h}} + \hat{l}\pi^{(4)}_{\hat{e}} - \hat{l}\pi^{(4)}_{e} - \hat{l}\pi^{(5)}_{\hat{e},\hat{h}-1} + (f_{\hat{s}} + 1 + n^R)\pi^{(5)}_{e,\hat{h}} - b_{\hat{s}}\pi^{(5)}_{e,\hat{h}-1}.$$

Note that, because a day off on day $\hat{h}$ is planned for employee e, the dual value $\pi^{(2)}_{e\hat{h}}$ is equal to 0 and the first term can be omitted.

Decision type d_3**:** For such a decision candidate q, the shift $\hat{s}$ of employee $\hat{e}$ is switched with a shift s of an employee e. Consequently, the time worked by employee $\hat{e}$ (resp. e) decreases by $l_{\hat{s}} - l_s$ (resp. $l_s - l_{\hat{s}}$). On day $\hat{h}$, the start and end times of the shifts assigned to employee $\hat{e}$ and e are also modified yielding the following cost increase estimate:

$$\Delta^q_{d_3} = \gamma_{d_3} + (l_{\hat{s}} - l_s)\pi^{(4)}_{\hat{e}} + (l_s - l_{\hat{s}})\pi^{(4)}_{e} + (f_s - f_{\hat{s}})\pi^{(5)}_{\hat{e},\hat{h}} - (b_s - b_{\hat{s}})\pi^{(5)}_{\hat{e},\hat{h}-1} + (f_{\hat{s}} - f_s)\pi^{(5)}_{e,\hat{h}} - (b_{\hat{s}} - b_s)\pi^{(5)}_{e,\hat{h}-1}.$$

Decision type $d_4(\eta)$**:** A candidate q of type $d_4(\eta)$ is a subset of m employees $e_1, \ldots, e_m$ assigned on day $\hat{h}$ to shifts $s_1, \ldots, s_m$, respectively, such that $m \leq \eta$, $w_{s_j} = w_{\hat{s}}$ for all $j \in \{1, \ldots, m\}$, $f_{s_j} < b_{s_{j+1}}$ for all $j \in \{1, \ldots, m-1\}$, $f_{s_1} > \hat{p}$, and $f_{s_m} < b_{\hat{s}}$. These employees are re-scheduled to new shifts $s^*_1, \ldots, s^*_m$, respectively. Thus, their work time decreases by $l_{s_j} - l_{s^*_j}$, $j \in \{1, \ldots, m\}$ and their shift start and end times are modified. Furthermore, the new shifts can yield an additional over-covering for job w in some periods. The set of these periods are denoted $\mathcal{P}^V_q$. For this candidate, the cost increase estimate is given by:

$$\Delta^q_{d_4} = \gamma_{d_4} + \hat{l}\pi^{(4)}_{\hat{e}} + \sum_{j=1}^{m} \left[(l_{s_j} - l_{s^*_j})\pi^{(4)}_{e_j} + (f_{s^*_j} - f_{s_j})\pi^{(5)}_{e_j,\hat{h}} - (b_{s^*_j} - b_{s_j})\pi^{(5)}_{e_j,\hat{h}-1}\right] + \sum_{p \in \mathcal{P}^V_q} \pi^{(5)}_{pw_{\hat{s}}}.$$

Decision type $d_5(\eta)$**:** For this type, a candidate q is an ordered subset of m employees $e_1, \ldots, e_m$ assigned on day $\hat{h}$ to shifts $s_1, \ldots, s_m$, respectively, such that $2 \leq m \leq \eta$, $w_{s_j} = w_{\hat{s}}$ for all $j \in \{1, \ldots, m\}$, $b_{s_j} \geq b_{\hat{s}}$ for all $j \in \{1, \ldots, m\}$, and $\max_{j \in \{1,\ldots,m\}} b_{s_j} \geq b_{\hat{s}} + \hat{l}$. Setting $e_0 = \hat{e}$ and $s_0 = \hat{s}$, shifts $s_0, \ldots, s_{m-1}$ are reassigned to employees $e_1, \ldots, e_m$, respectively, and shift s_m to employee e_0 (assuming $b_{s_m} \geq b_{\hat{s}} + \hat{l}$). Therefore, the estimate of the cost increase for candidate q is equal to:

$$\Delta^q_{d_5} = \gamma_{d_5} + \sum_{j=0}^{m} \left[(l_{s_j} - l_{s_{j-1}})\pi^{(4)}_{e_j} + (f_{s_{j-1}} - f_{s_j})\pi^{(5)}_{e_j,\hat{h}} - (b_{s_{j-1}} - b_{s_j})\pi^{(5)}_{e_j,\hat{h}-1}\right],$$

where $j-1$ is assumed to be equal to m when $j = 0$.

Given that the number of candidates for decision type $d_5(\eta)$ can be large, the problem of finding the best decision of this type can be formulated as a network flow problem. Let $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ be a network, where $\mathcal{N}$ and $\mathcal{A}$ are its node and arc sets, respectively. In $\mathcal{N}$, there is a node for $e_0 = \hat{e}$ and for each employee e that is scheduled to work a shift $s \in \mathcal{S}^{\hat{h}}_e$ on day $\hat{h}$ such that $w_s = w_{\hat{s}}$ and $b_s \geq b_{\hat{s}}$. Let n^N be the number of these latter employees that we denote e_i, $i \in \{1, \ldots, n^N\}$. Their respective shifts are denoted s_i, $i \in \{1, \ldots, n^N\}$. In $\mathcal{A}$, there exists an arc between every pair of nodes e_i and e_j if employee e_i can be assigned to shift s_j. This arc (e_i, e_j) represents the re-scheduling of employee e_i to the shift s_j and bears a cost

$$c_{ij} = (l_{s_i} - l_{s_j})\pi^{(4)}_{e_i} + (f_{s_j} - f_{s_i})\pi^{(5)}_{e_i,\hat{h}} - (b_{s_j} - b_{s_i})\pi^{(5)}_{e_i,\hat{h}-1}.$$

Figure 3 shows an example of a network $\mathcal{G}$ where $n^N = 4$. This network is almost complete: only the arcs (e_0, e_1) and (e_4, e_2) are missing. These arcs may not exist because, for example, shift s_1 begins too early for the late employee e_0 and shift s_2 ends too late for employee e_4 in order to respect the minimum rest time between days $\hat{h}$ and $\hat{h}+1$.

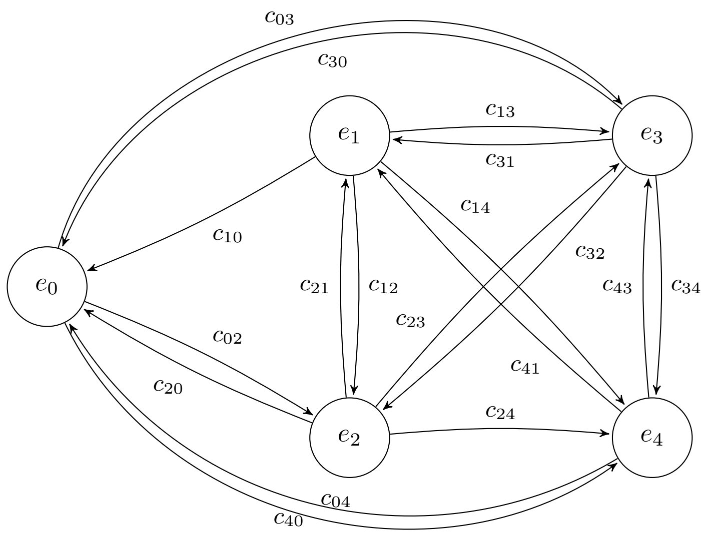

Figure 3: Exemple of a network (with $n^N = 4$) used for decision type $d_5(\eta)$

Given a network $\mathcal{G}$, the problem of finding the best decision of type $d_5(\eta)$ consists of finding a least-cost elementary circuit passing through node e_0 and containing at most η arcs. This problem can be solved using dynamic programming or integer programming. For our tests, we chose integer programming. The computed optimal circuit indicates the employees involved in the decision and the switching order of their corresponding shifts. The cost increase estimate is given by the sum of the costs of the arcs forming this circuit plus the managerial cost γ_{d_5}.

In Algorithm 1, decision type $d_5(\eta)$ is, therefore, treated differently than the other types and omitted from the loop in Step 3. After this loop, the integer linear program based on network $\mathcal{G}$ is constructed and solved using a commercial mixed-integer programming solver. The cost increase estimate is then compared to the best one found for the other types.

It should be noted that the the minimum rest time constraints (5) are often not binding in practice and their corresponding dual values $\pi^{(5)}_{e,h}$ are, thus, often equal to 0. To speed up the proposed heuristic, we have decided to omit the terms involving these dual values in all cost increase estimates presented above.

3.2 A sequence of disruptions

To solve the personnel re-scheduling problem, we proposed in the previous section a heuristic that is based on the dual values obtained when solving the personnel scheduling problem, that is, when planning the initial schedule. Typically, more than one minor disruption occurs during a week. Hence, we can apply the heuristic described in Algorithm 1 for each disruption as they occur. However, when solving the personnel re-scheduling problem for the second and subsequent disruptions, the dual values derived from the initial personnel scheduling problem are not accurate anymore given that the schedules of some employees have changed when resolving the previous disruptions. In fact, the dual values should be updated after each disruption so that they can reflect the schedule changes. Solving an updated personnel scheduling problem might not be applicable, for example, when there is not enough time between two disruptions. Furthermore, this personnel scheduling problem would require additional constraints to enforce the current schedule, yielding a very different dual solution. In this section, we propose a simple procedure to update the dual values without solving a personnel scheduling problem.

The formulas used to estimate the cost increase of each candidate of each decision type involve the dual variables $\pi_e^{(4)}$, $e \in \mathcal{E}$, and $\pi_{pw}^{(5)}$, $p \in \mathcal{P}$, $w \in \mathcal{W}$. Observe, however, that for a given minor disruption $(\hat{e}, \hat{h}, \hat{s}, \hat{l}, \hat{p})$, not all decisions may yield additional over-coverings and, if this occurs, the additional over-coverings take place in between periods $\hat{p}$ and $b_{\hat{s}}$ for task $w_{\hat{s}}$. Consequently, most probably, only the dual values $\pi_{pw}^{(5)}$ for a period $p \in [\hat{p}+1, b_{\hat{s}-1}]$ and task $w_{\hat{s}}$ can be affected by this disruption. Given that most subsequent disruptions will become known later than $b_{\hat{s}}$ or will involve a task $w \neq w_{\hat{s}}$, there is no need to update the dual values $\pi_{pw}^{(5)}$, $p \in \mathcal{P}$, $w \in \mathcal{W}$. The proposed procedure, thus, concentrates on the dual values $\pi_e^{(4)}$, $e \in \mathcal{E}$.

A statistical study allowed us to assume that there exists a real function ψ such that, for each employee $e \in \mathcal{E}$, we have $\pi_e^{(4)} \backsimeq \psi(L_e)$, where $L_e = \sum_{k \in \mathcal{K}^L} L_e^k$ is the total time worked by employee e in the current solution. We propose to find a function $\tilde{\psi}$ to approximate function ψ. Given that the shape of ψ is unknown, we resort to a nonparametric regression method to determine $\tilde{\psi}$, namely, the *Multivariate Adaptive Regression Splines* (MARS) method introduced by Friedman [8] in 1991. To the best of our knowledge, this method has never been used to update dual values. To approximate a real function $f(x_1, \ldots, x_n)$ of one or several independent variables, the MARS method assumes that the approximate function $\tilde{f}$ belongs to a vectorial space generated by a set of basis functions. These basis functions can take the form of a constant, a linear spline ($x \mapsto (x-t)_+$ where t is a constant and the $+$ subscript indicates a zero value for a negative argument), a quadratic spline ($x \mapsto (x-t)_+^2$) and a product of these functions. This vectorial space is larger than the one used for parametric regression methods, which may explain its efficiency with respect to classical regression methods. The MARS method determines a model for the approximate function $\tilde{f}$ in two phases. First, it selects iteratively the basis functions. Second, it eliminates iteratively the components that do not have a large impact on the estimation error.

In our case, to find an approximate function $\tilde{\psi}$, we consider the set of sample points $\{(L_e, \pi_e^{(4)}) \mid e \in \mathcal{E}\}$ obtained from the linear relaxation of the initial personnel scheduling problem. Then, we apply the MARS method on this set of points to yield a function $\tilde{\psi}$ such that $\pi_e^{(4)} \backsimeq \tilde{\psi}(L_e)$ for all $e \in \mathcal{E}$. Given this approximate function, the dual values are updated before resolving each disruption occurring during the same week except the first one. The decision made to resolve disruption k modifies the schedules of a subset of employees denoted $\mathcal{E}^k$ and incurs a variation of λ_e^k in the total number of periods worked by employee $e \in \mathcal{E}^k$. Before resolving each disruption $k \geq 2$, the dual value update is performed as follows:

$$\left(\pi_e^{(4)}\right)^k = \begin{cases} \tilde{\psi}(L_e + \sum_{i=1}^{k-1} \lambda_e^i) & \text{if } e \in \mathcal{E}^{k-1} \\ \left(\pi_e^{(4)}\right)^{k-1} & \text{otherwise,} \end{cases}$$

where $\left(\pi_e^{(4)}\right)^{k-1}$, $e \in \mathcal{E}$, are the dual values used to solve disruption $k-1$ and $\left(\pi_e^{(4)}\right)^1 = \pi_e^{(4)}$, $e \in \mathcal{E}$.

4 Computational experiments

To assess the efficiency of the re-scheduling heuristic described in Algorithm 1 and of the procedure updating the dual values introduced in Section 3.2, we conducted intensive computational experiments. The results of these experiments are reported in Section 4.2 for the single-disruption case and in Section 4.3 for the multiple-disruption case. Beforehand, we described in Section 4.1 the datasets used for these tests and how the disruption scenarios were generated.

All algorithms were implemented in C++ and all integer programs were solved using the CPLEX MIP solver, version 12.6.1.0. All tests were executed on a Linux machine equipped with an 8-core Intel Core i7 processor clocked at 3.4GHz and a RAM of 16Gb.

4.1 Datasets and disruption scenarios

To perform our tests, we use real-world datasets provided by our industrial partner. The characteristics of these data sets are given in Table 1, together with the integrality gaps obtained by solving model (1)–(11).

These gaps are very small and support our assumption that the dual values derived from the linear relaxation solution carry valuable information with respect to the integer solution. All these data sets consider a one-week horizon divided into 15-minute periods.

Table 1: Characteristics of the datasets

Dataset	No. employees	No. jobs	Integrality gap (%)
I_1	15	5	0
I_2	25	5	$< 10^{-3}$
I_3	29	6	0
I_4	32	3	$< 10^{-3}$
I_5	49	7	$< 10^{-3}$
I_6	95	7	$< 10^{-3}$
I_7	191	44	$< 10^{-3}$

For the single-disruption case, our tests focused on one decision type at a time. For each type $d \in \mathcal{D}$ and each dataset I_k, $k \in \{1, \dots, 7\}$, 30 minor disruptions $(\hat{e}, \hat{h}, \hat{s}, \hat{l}, \hat{p})$ were generated as follows. First, day $\hat{h}$ is selected using a uniform distribution over $\mathcal{H}$. Second, employee $\hat{e}$ is drawn from a uniform distribution over the set of employees working on day $\hat{h}$. Shift $\hat{s}$ corresponds to the shift worked by employee $\hat{e}$ on day $\hat{h}$. Third, the length of the lateness $\hat{l}$ in number of periods is chosen using another uniform distribution in the interval $[2, l_{\hat{s}} - 4]$. Fourth, the period $\hat{p}$ at which the manager learns that employee $\hat{e}$ will be late is drawn from a uniform distribution in the interval starting at the beginning of day $\hat{h}$ and ending at period $b_{\hat{s}} - 1$ if $d \in \{d_1, d_4(\eta)\}$. For the other three decision types, period $\hat{p}$ does not impact on the optimal decisions as long as it provides enough time for an employee to reach work from home. Finally, the set of candidates $\mathcal{Q}_d(\hat{e}, \hat{h}, \hat{s}, \hat{l}, \hat{p})$ is built. If it contains less than three candidates, the disruption is rejected because the corresponding re-scheduling problem is considered too simple and, after removing employee $\hat{e}$ from the possible choices on day $\hat{h}$, the process is repeated from the second step to find a replacing disruption on day $\hat{h}$.

For the multiple-disruption case, we also consider one decision type at a time, i.e., all disruptions will be resolved using the same decision type. For each type $d \in \mathcal{D}$ and each dataset I_k, $k \in \{1, \dots, 7\}$, we generated 30 sequences of minor disruptions. To determine each sequence, we first draw a number of potential disruptions n^D from a uniform distribution in $[2, 50]$. Then, for each $j \in \{1, \dots, n^D\}$, we try to generate a minor disruption as in the single-disruption case described above, except that there is no repetition if the disruption is rejected, i.e., if the candidate set contains less than three candidates. Every time that a disruption $(\hat{e}, \hat{h}, \hat{s}, \hat{l}, \hat{p})$ is accepted, we force the next disruption to occur later than at period $b_{\hat{s}}$ by removing from the possible choices all days $h < \hat{h}$ and, if day $\hat{h}$ is drawn again, all employees e working a shift s on day $\hat{h}$ such that $b_s \leq b_{\hat{s}}$.

4.2 Computational results for the single-disruption case

As mentioned above, we tested each decision type $d \in \mathcal{D}$ separately. This allows to see if the cost increase estimate computed by the re-scheduling heuristic is accurate for each type individually. For each decision type, 30 disruptions are generated for each dataset, yielding a total of 210 test instances for each type. For decision type $d_4(\eta)$, we set $\eta = 2$ because it is difficult to find more than two employees in a candidate. For type $d_5(\eta)$, we chose to set $\eta = \left|\mathcal{Q}_{d_3}(\hat{e}, \hat{h}, \hat{s}, \hat{l}, \hat{p})\right|$ for a disruption $(\hat{e}, \hat{h}, \hat{s}, \hat{l}, \hat{p})$, allowing the largest flexibility in this case. Note that, because we do not compare decisions of different types, the managerial costs γ_d, $d \in \mathcal{D}$, are irrelevant and have been set to 0.

Consider a decision type $d \in \mathcal{D}$ and an instance for this type whose disruption is defined by $(\hat{e}, \hat{h}, \hat{s}, \hat{l}, \hat{p})$. Let $q^* \in \mathcal{Q}_d(\hat{e}, \hat{h}, \hat{s}, \hat{l}, \hat{p})$ be a candidate yielding the least cost increase estimate $\Delta_d^{q^*}$ for this instance. To assess the quality of the corresponding decision, we will compare it to the decisions yielded by two other methods for addressing the same disruption, namely, a greedy and an exact method.

The greedy method corresponds to a simple procedure that might be used in practice by a manager who does not rely on an algorithm to resolve a disruption. The candidate $q^G \in \mathcal{Q}_d(\hat{e}, \hat{h}, \hat{s}, \hat{l}, \hat{p})$ returned by this method depends on the decision type:

d_1: Choose the candidate (an employee) whose shift ends the latest before $b_{\hat{s}}$. If several candidates end at the same period, then choose the one with the least working time during the horizon.

d_2: Choose the candidate (an employee having a day off) with the least working time during the horizon.

d_3: Choose the candidate whose shift begins the earliest after $b_{\hat{s}} + \hat{l} - 1$, i.e., the last period of the uncovered demand block.

$d_4(\eta)$: Choose the same candidate as for type d_1.

$d_5(\eta)$: Choose the same candidate as for type d_3.

The exact method is based on the assumption that the disruption is resolved by modifying shifts on day $\hat{h}$ only but the manager may revise the employee schedules for the rest of the horizon at the end of day $\hat{h}$. Indeed, this often occurs in practice because the decision made in real time to address a disruption may bring an employee in overtime if he/she works the rest of his/her scheduled shifts. The manager may want to cut this overtime as much as possible. In this case, schedule changes affecting one or several employees are applied. To fully test the robustness of the decisions proposed by our re-scheduling heuristic, we assume for our tests that these changes might impact all employees. Consequently, the exact method consists of solving a personnel scheduling problem (through model (1)–(11)) restricted as follows. All planned shifts scheduled on a day prior to $\hat{h}$ are fixed. On day $\hat{h}$, all planned shifts starting before period $\hat{p}$ are also fixed, while the others are considered except shift $\hat{s}$. On day $\hat{h}$, we also consider all shifts associated with a candidate in $\mathcal{Q}_d(\hat{e}, \hat{h}, \hat{s}, \hat{l}, \hat{p})$. Finally, all shifts, planned or not, are considered for the days after day $\hat{h}$.

To compare fairly the decisions proposed by the three methods, we produced two new solutions, one from each decision imposed by the re-scheduling heuristic and the greedy heuristic. These new solutions were obtained by fixing the corresponding decision on day $\hat{h}$ and by re-optimizing the rest of the week as in the exact method. Let σ^i, $i = R, G, E$, be the solutions derived from the re-scheduling heuristic ($i = R$), the greedy heuristic ($i = G$), and the exact method ($i = E$). Furthermore, for $i = R, G, E$, let $z_d(\sigma^i)$ be the cost of solution σ^i and ζ_d^i be the cost increase between $z_d(\sigma^i)$ and the cost of the initial planned solution.

Given the large number of results obtained (3 methods, 5 decision types and 210 instances for each method and type), we present them in an aggregate form which allows to highlight the performance of the re-scheduling heuristic for each decision type. For each decision type $d \in \mathcal{D}$, we compare the values ζ_d^R, ζ_d^G, and ζ_d^E for each instance and classify this instance into one of the three following categories. If $\zeta_d^R = \zeta_d^E$, we say that the decision proposed by re-scheduling heuristic is *optimal* for this instance. If $\zeta_d^E < \zeta_d^R \leq \zeta_d^G$, we say that the re-scheduling heuristic obtained a *success*. Otherwise ($\zeta_d^R > \zeta_d^G$), the result is categorized as a *failure*.

The results by decision type and success category are reported in Figure 4, where the last subfigure also gives the results when aggregating all decision types. For each decision type $d \in \mathcal{D}$, the average number of candidates considered per instance, denoted μ_d, is also indicated and varies between 4.7 and 6×10^7. The results are impressive: the re-scheduling heuristic computes an optimal decision for more than 95% of the instances and yields a failure in less than 2% of the cases. We observed that a large number of failures occur for instances based on dataset I_3. For this dataset, the dual values $\pi_e^{(4)}$, $e \in \mathcal{E}$, are more or less the same for all employees working more than 32 hours per week. Hence, they are not always sufficient to identify the right decision when several of these employees are part of several candidates. On the other hand, the observed computational times for the re-scheduling heuristic are sufficiently small to use this heuristic in real time. Indeed, for the 1050 instances, the average computational time is less than one second. The computational times for the greedy heuristic are also very small, while the exact method can require computational times that are much larger for certain instances (more than 10 minutes in some cases).

To better compare the re-scheduling heuristic with the greedy heuristic, we also computed, for each decision type $d \in \mathcal{D}$, the pourcentage of the instances for which $\zeta_d^R < \zeta_d^G$. The results are presented in Table 2. We observe that, for the decision types d_1, d_2 and d_4, the re-scheduling heuristic outperforms the greedy heuristic for a relatively small number of instances but not negligible. However, for the other two types (d_3 and d_5), the former heuristic clearly dominates the latter. It should be noted that, when the solution provided by the greedy heuristic is worse than the one computed by the re-scheduling heuristic, it can be of very poor quality, while the opposite rarely occurs. The results reported in the next section will support this statement.

Table 2: Pourcentage of instances for which $\zeta_d^R < \zeta_d^G$ for each decision type d

d_1	d_2	d_3	d_4	d_5
15.7	11.4	36.7	14.8	47.6

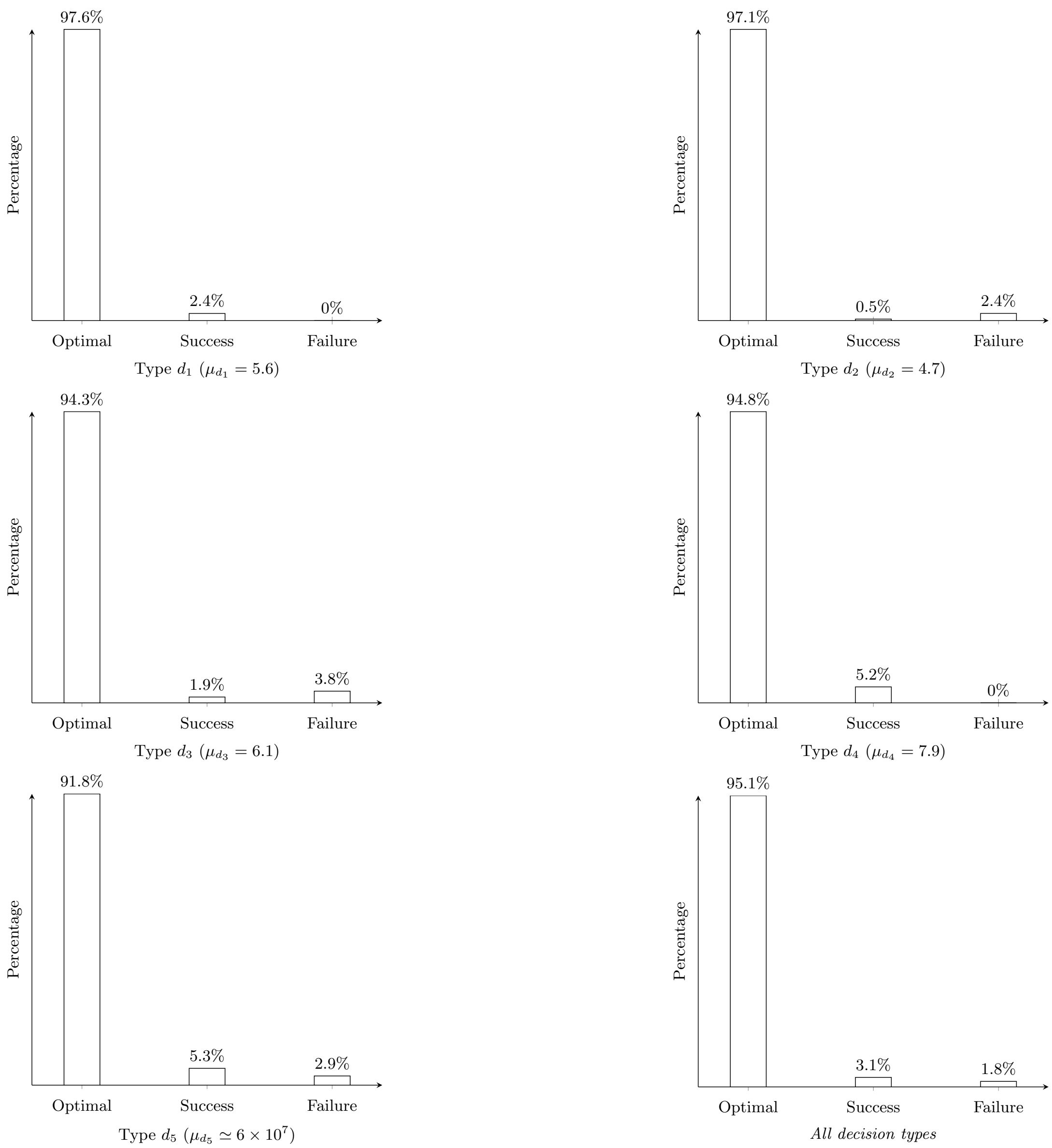

Figure 4: Success rate by category of success for each decision type

Finally, in Figure 5, we report additional results to evaluate if the ranking of the decisions performed by the re-scheduling heuristic based on the cost increase estimate corresponds to the ranking of the costs obtained after re-optimizing the schedule for the rest of the week. These results focus on the 30 instances tested for decision type $d_4(2)$ and dataset I_6 (similar results were observed for the other instances and datasets). For each of these instances (numbered from 1 to 30 on the horizontal axis), we indicate the costs $z(\sigma_j^R)$, $j = 1, 2, 3$, of the three solutions σ_j^R which are derived from the best three candidates (in this order) identified by this heuristic. Observe that, for all instances except the third one, the order of the decisions is preserved, that is, $z(\sigma_{j_1}^R) \leq z(\sigma_{j_2}^R)$ if $j_1 < j_2$. Consequently, the cost increase estimate seems to be positively correlated with the decision quality.

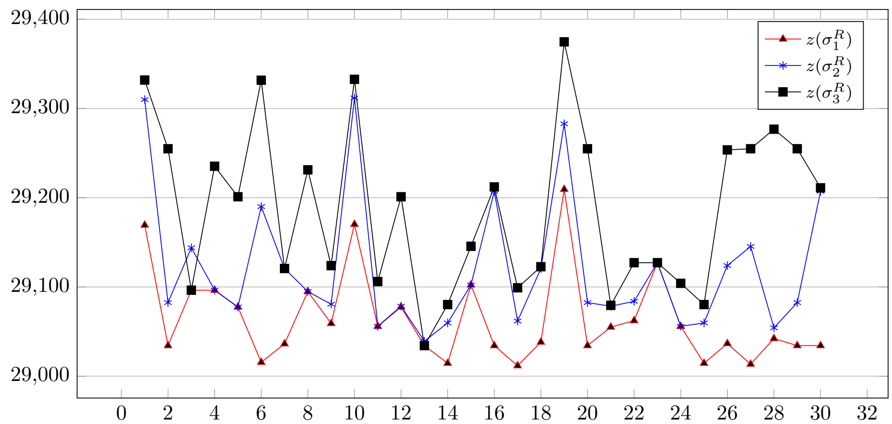

Figure 5: Cost of the three best solutions derived from the re-scheduling heuristic for decision type $d_4(2)$

4.3 Computational results for the multiple-disruption case

For the multiple-disruption case, we compare three algorithms, namely, the re-scheduling heuristic without updating the dual values, the re-scheduling heuristic with an update of the dual values after each disruption, and the greedy heuristic. For each decision type and each instance, the experiment unfolds as follows. For each disruption (treated in chronological order), the three heuristics and the exact method are applied to propose a decision each. Three solutions derived from the decisions of the three heuristics are computed as in the single-disruption case by fixing the decision for the disruption day and by re-optimizing the end of the week as in the exact method. These solutions and that of the exact method are denoted σ^i, $i = R, U, G, E$ (U for re-scheduling with dual value updates). Their costs are denoted by $z_d(\sigma^i)$, $i = R, U, G, E$, and we compute the nonnegative cost difference $z_d(\sigma^i) - z_d(\sigma^E)$, $i = R, U, G$, of each heuristic solution with respect to the exact solution. Next, the shift modifications on the day of the disruption suggested by the exact method are implemented before moving on to the next disruption. We denote by Γ_d^i, $i = R, U, G$, the sum of the cost differences incurred over all disruptions by the re-scheduling heuristic without and with dual value updates and the greedy heuristic, respectively.

Recall that, for each decision type and each dataset, 30 instances were generated. These instances differ by the sequence of disruptions considered. The results of our experiments are reported in Table 3. For each dataset and each decision type, this table specifies the average and the maximum number of disruptions per instance (n_{avg}^D and n_{max}^D), the average and the maximum computational time (T_{avg} and T_{max}) in seconds to resolve one disruption by the re-scheduling heuristic with dual value updates (including the time to update the dual values), the average sum of the cost differences obtained by the re-scheduling heuristic with and without dual value updates (Γ_d^R and Γ_d^U), and the average sum of the cost differences of the solutions computed by the greedy heuristic (Γ_d^G).

From these results, we make the following observations. The number of disruptions per instance tends to increase with the dataset size (number of employees). This is due to the rule that a disruption is accepted only

Table 3: Results for the multiple-disruption case

Dataset	Decision type	n^D_{avg}	n^D_{max}	Re-scheduling T_{avg} (s)	T_{max} (s)	Γ^R_d	Γ^U_d	Greedy Γ^G_d
I_1	d_1	2.7	4	0.07	0.12	0	0	0
	d_2	3	3	0.03	0.06	0	0	0
	d_3	2	2	0.05	0.07	0.3	1	13.9
	d_4	2.6	4	0.05	0.09	0	0	0
	d_5	2.3	3	0.05	0.11	5.1	5.7	11.4
I_2	d_1	2.2	4	0.06	0.09	0	0	0.8
	d_2	6.1	9	0.02	0.03	22.2	0	0
	d_3	2.1	4	0.04	0.05	1.4	0.7	44.1
	d_4	2.2	4	0.04	0.10	0	0	0.8
	d_5	3.3	5	0.05	0.17	8.5	4.9	38.3
I_3	d_1	2.4	4	0.19	0.21	0	0	0
	d_2	7.9	12	0.19	0.21	0	0	0
	d_3	2.3	5	0.23	0.25	8.6	1.5	2.8
	d_4	2.1	3	0.21	0.27	3.2	3.6	4.1
	d_5	5.3	8	0.20	0.30	7	1.2	4
I_4	d_1	2.5	6	0.10	0.15	0	0	0
	d_2	5	7	0.05	0.07	4.5	0	1.5
	d_3	2.7	5	0.07	0.10	0.1	0.1	4.4
	d_4	4.9	10	0.08	0.19	1.8	0.9	2.8
	d_5	4.6	8	0.09	0.21	2.3	1.2	6
I_5	d_1	2.2	4	0.18	0.27	0.2	0.2	0.2
	d_2	4.7	7	0.15	0.17	3.5	0	0.1
	d_3	3.5	7	0.21	0.22	4.5	3.0	22.9
	d_4	2.2	4	0.21	0.29	0.2	0.1	0.2
	d_5	6.1	11	0.24	0.38	11.9	11.3	33.5
I_6	d_1	2.4	4	0.29	0.35	1.9	1.1	2.9
	d_2	17.4	24	0.11	0.13	26.7	8.1	37
	d_3	6.1	13	0.20	0.23	4.7	2.5	56.6
	d_4	2.7	5	0.33	0.61	0.7	0	2.9
	d_5	9.6	15	0.34	0.74	30.7	11.6	56.5
I_7	d_1	7	14	1.42	1.59	11.8	3.1	18.7
	d_2	27.1	37	0.51	0.64	65.9	8.0	8.1
	d_3	8.5	18	1.22	1.29	21.1	4.5	38.7
	d_4	7.6	14	1.64	2.12	7.0	1.1	20.0
	d_5	17.9	28	1.54	2.29	33.3	4.1	45.3

if there are at least three candidates to resolve it. Concerning the computational times of the re-scheduling heuristic with dual value updates, the results show that they are very acceptable, reaching a largest average time of 1.64 seconds for datasets involving up to 191 employees. Now, comparing the values of Γ^R_d and Γ^U_d, we deduce that updating the dual values can be highly useful in several cases, especially for decision type d_5 for which a decision may impact several employees at the same time and, thus, there is a relatively high probability that the schedule of the same employee be modified more than once during the week. We also observe a large impact for decision type d_2 where the number of hours worked by the employee who is asked to work in a day off may change considerably. On the other hand, in certain cases (for instance, dataset I_1 and type d_5), updating the dual values may slightly deteriorate the solution quality. This is not surprising given that we are comparing heuristics. Nevertheless, the improvements obtained for several cases (for instance, dataset I_2 and type d_2 or dataset I_7 and type d_5) are substantial.

Next, let us compare the values of Γ^U_d and Γ^G_d. In all cases, we get $\Gamma^U_d \leq \Gamma^G_d$, showing the superiority of the re-scheduling heuristic with dual value updates over the greedy heuristic. In certain cases, the value of Γ^G_d is much larger than the value of Γ^R_d, indicating that the greedy heuristic can compute bad-quality solutions in some cases. Note, however, that without updates, the re-scheduling heuristic does not always yield a better average cost difference Γ^R_d than that of the greedy heuristic Γ^G_d. Finally, we observe that the average cost difference with respect to the cost increase of the exact method is relatively small (recall that, for dataset I_6, the total costs are around 29,100 according to Figure 5). This can be explained by the results obtained for the single-disruption case which showed that the re-scheduling heuristic returns the optimal decision most of the times. On average, the sum of the cost differences decreases by 73.3% when updating the dual values between consecutive disruptions.

5 Conclusion

In this paper, we considered the personnel re-scheduling problem that must be solved in real time after the occurrence of a minor disruption in a context where the employees can be assigned to a wide variety of shifts, starting and ending at various times. To solve this problem, we propose a fast re-scheduling heuristic that can handle five different types of decision. This heuristic is based on the dual values obtained when solving in the planning phase the linear relaxation of the personnel scheduling problem. The computational results obtained on 1050 instances derived from real-world datasets involving up to 191 employees showed the efficiency of this re-scheduling heuristic. It computed in less than one second on average an optimal solution for more than 95% of these instances. We also developed a procedure for updating the dual values after each disruption when several disruptions must be dealt with during the same week. Additional computational results showed the usefulness of this updating procedure which reduced the sum of the gaps between the optimal value and the heuristic solution value by more than 73% in our tests.

Several research avenues can be pursued after this work, including the following two. First, in certain companies, the planned personnel schedules are not necessarily computed by solving a mixed integer program. They are rather determined manually from past schedules. In this case, no dual values are available and designing an algorithm to determine accurate dual values would be a valuable complement to our work. Second, the proposed re-scheduling heuristic is myopic in the sense that it considers only the current disruption without anticipating future disruptions. Developing a stochastic personnel re-scheduling algorithm that evaluate the expected future costs for each possible decision constitutes a challenging research topic.

References

[1] Bard, J. F., Purnomo, H. W., 2005. Hospital-wide reactive scheduling of nurses with preference considerations. IIE Transactions 37(7), 589–608.

[2] Burke, E., De Causmaecker, P., Vanden Berghe, G., 2004. The state of the art of nurse rerostering. Journal of Scheduling 7(6), 441–499.

[3] Cacchiani, V., Huisman, D., Kidd, M., Kroon, L., Toth, P., Veelenturf, L., Wagenaar, J., 2014. An overview of recovery models and algorithms for real-time railway rescheduling. Transportation Research Part B 63(1), 15–37.

[4] Clausen, J., Larsen, J., Rezanova, N., 2010. Disruption management in the airline industry – Concepts, models and methods. Computers and Operations Research 37(5), 809–821.

[5] Dantzig, G. B., 1954. Letter to the editor – A comment on Edie's Traffic delays at toll booths. Journal of the Operations Research Society of America 2(3), 339–341.

[6] Edie, L. C., 1954. Traffic delays at toll booths. Journal of the Operations Research Society of America 2(2), 107–138.

[7] Ernst, A. T., Jiang, H., Krishnamoorthy, M., Owens, B., Sier, D., 2004. An annotated bibliography of personnel scheduling and rostering. Annals of Operations Research 127(1-4), 21–144.

[8] Friedman, J. H., 1991. Multivariate adaptive regression splines. Annals of Statistics, 1–67.

[9] Froger, C., 2015. Mise à jour des horaires de personnel travaillant sur des quarts. Master's dissertation, Polytechnique Montréal. In French.

[10] Kitada, M., Morizawa, K., 2013. A heuristic method for nurse rerostering problem with a sudden absence for several consecutive days. International Journal of Emerging Technology and Advanced Engineering 3(11), 353–361.

[11] Mahenhout, B., Vanhoucke, M., 2011. An evolutionary approach for the nurse rerostering problem. Computers & Operations Research 38(10), 1400–1411.

[12] Moz, M., Pato, M. V., 2003. An integer multicommodity flow model applied to the rerostering of nurse schedules. Annals of Operations Research 119(1-4), 285–301.

[13] Moz, M., Pato, M. V., 2004. Solving the problem of rerostering nurse schedules with hard constraints: New multicommodity flow models. Annals of Operations Research 128(1–4), 179–197.

[14] Moz, M., Pato, M. V., 2007. A genetic algorithm approach to a nurse rerostering problem. Computers & Operations Research 34(3), 667–691.

[15] 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.

[16] 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.