Zuse Institute Berlin

Takustr. 7
14195 Berlin
Germany

THOMAS BREUGEM
CHRISTOF SCHULZ
THOMAS SCHLECHTE
RALF BORNDÖRFER

A Three-Phase Heuristic for Cyclic Crew Rostering with Fairness Requirements

ZIB Report 19-43 (August 2019)

Zuse Institute Berlin
Takustr. 7
14195 Berlin
Germany

Telephone: +49 30-84185-0
Telefax: +49 30-84185-125

E-mail: bibliothek@zib.de
URL: http://www.zib.de

ZIB-Report (Print) ISSN 1438-0064
ZIB-Report (Internet) ISSN 2192-7782

A Three-Phase Heuristic for Cyclic Crew Rostering with Fairness Requirements*

Thomas Breugem[1], Christof Schulz[2], Thomas Schlechte[2], Ralf Borndörfer[3]

[1]Econometric Institute and Erasmus Center for Optimization in Public Transport
Erasmus University Rotterdam, The Netherlands

[2]LBW Optimization GmbH
Berlin, Germany

[3]Zuse Institute Berlin
Berlin, Germany

breugem@ese.eur.nl, {schulz,schlechte}@lbw-optimization.de, borndoerfer@zib.de

Abstract

In this paper, we consider the Cyclic Crew Rostering Problem with Fairness Requirements (CCRP-FR). In this problem, attractive cyclic rosters have to be constructed for groups of employees, considering multiple, a priori determined, fairness levels. The attractiveness follows from the structure of the rosters (e.g., sufficient rest times and variation in work), whereas fairness is based on the work allocation among the different roster groups. We propose a three-phase heuristic for the CCRP-FR, which combines the strength of column generation techniques with a large-scale neighborhood search algorithm. The design of the heuristic assures that good solutions for all fairness levels are obtained quickly, and can still be further improved if additional running time is available. We evaluate the performance of the algorithm using real-world data from Netherlands Railways, and show that the heuristic finds close to optimal solutions for many of the considered instances. In particular, we show that the heuristic is able to quickly find major improvements upon the current sequential practice: For most instances, the heuristic is able to increase the attractiveness by at least 20% in just a few minutes.

Keywords: Crew Planning, Column Generation, Variable-Depth Neighborhood Search

*Parts of this work have been developed within the Research Campus MODAL (Mathematical Optimization and Data Analysis Laboratories) funded by the German Ministry of Education and Research (BMBF).

1 Introduction

The scheduling of personnel is one of the most challenging planning problems for a public transport operator. This is partly due to the large-scale nature of the problem, but also due to the two conflicting objectives: On the one hand, the operator must minimize cost from an operational point of view, yet on the other hand the operator must also maximize the quality of work from an employees' point of view. The Netherlands, for example, has a history of strikes from employees of Netherlands Railways (NS), expressing their discontent regarding the scheduled work. Reoccurring themes during these conflicts are, for example, the irregularity of scheduled work, and the distribution of work among the different crew bases. Hence, it is clear that, in order to avoid such conflicts, a public transport operator should incorporate the demands of employees in the planning process.

The assignment of work to the employees is traditionally decomposed into crew scheduling and crew rostering. In the former, the duties (i.e., working days) are constructed, and, in the latter, these duties are assigned to the employees. The focus in crew rostering lies on *perceived fairness* and *perceived attractiveness*, leading to a bi-objective decision problem. This problem was formalized in Breugem et al. [2017] as the Fairness-oriented Crew Rostering Problem (FCRP). Roughly speaking, allocations are perceived fair whenever each crew member performs similar work (measured over a given number of attributes). Perceived attractiveness, on the other hand, focuses on the structure of the rosters, taking, for example, the workload in each week and the rest time between consecutive working days into account. Although both objectives have overlapping components, there is a clear trade-off, as shown in Breugem et al. [2017].

It is evident that perceived fairness and attractiveness of rosters are far more ambiguous concepts than, for example, the cost of a rolling stock schedule. As a result, the crew rostering problem is generally solved multiple times for varying parameter settings, thereby steering towards a desired solution. This implies that, even during the tactical planning phase, it is desirable that high quality solutions can be obtained quickly. Furthermore, exact methods can be intractable for some of the instances encountered in practice: The exact approach developed in Breugem et al. [2017] is able to solve instances of about three roster groups and 100 duties (obtained from crew base Utrecht) in at most a few hours, but fails to obtain good solutions for instances of, say, six groups and about 200 duties in reasonable time. The latter is the size of crew base Amersfoort, where NS conducted a pilot study regarding decision support for crew rostering in the beginning of 2018. The pilot sparked interest and enthusiasm for decision support, but also highlighted the need for a heuristic algorithm for the larger instances, and is therefore a driving force behind this work.

In this paper, we consider a variant of the FCRP, which we will call the Cyclic Crew Rostering Problem with Fairness Requirements (CCRP-FR). This problem deviates from

the FCRP by assuming a fixed, a priori known, set of fairness levels for which rosters must be constructed. This differs from the FCRP, where we aim at finding the entire trade-off curve and hence determining the relevant fairness levels is part of the solution process. In practice, the former setting is often encountered, as planners generally have some fairness levels (e.g., high, medium, and low fairness) in mind for which they want to construct rosters. One key difference with the FCRP is that the solution with maximum fairness does not necessarily have to be computed, which can be time consuming for large instances.

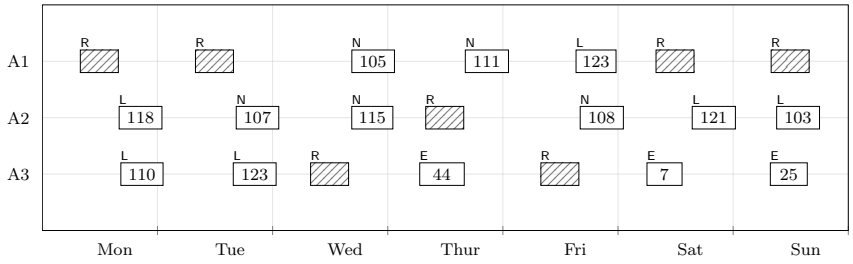
The main contribution of this paper is a three-phase heuristic for the CCRP-FR, which combines the strengths of the exact approach for the FCRP developed in Breugem et al. [2017] with a large-scale neighborhood search algorithm. The design of the heuristic assures that good solutions for all fairness levels are obtained quickly, and can still be further improved if additional running time is available. We evaluate the performance of the proposed solution approach using real-world instances from NS. In particular, we show that the three-phase heuristic finds close to optimal solutions for most instances, and achieves a major improvement (up to 40%) over the current (sequential) approach.

The remainder of this paper is organized as follows. In Section 2 we discuss the CCRP-FR in detail, and in Section 3, we give an overview of related work. In Section 4 we discuss the row-based formulation, followed by a detailed description of the three-phase heuristic in Section 5. Section 6 evaluates the performance of the solution method on practical instances from NS, and the paper is concluded in Section 7.

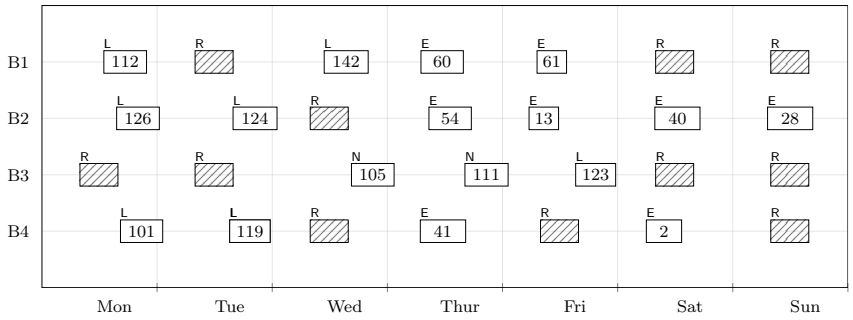
2 Cyclic Crew Rostering with Fairness Requirements

The goal of crew rostering is to construct rosters for the employees whilst taking both perceived fairness and perceived attractiveness into account. The crew is partitioned into *roster groups*, and each of these groups has to be assigned a roster. Figure 1 shows an example of two possible rosters, one for group A, consisting of three employees, and one for group B, consisting of four employees.

The roster groups operate in *cyclic rosters*, i.e., the rosters are executed by multiple employees in a periodic fashion. These rosters are constructed for a period of one year, and the work for this period is assumed to be cyclic. In other words, Monday the 22th of April is identical to Monday the 29th of April. Each roster has an underlying structure, known as the *basic schedule*. The basic schedule specifies the type of work of each day (e.g., a late duty or day-off). Each basic schedule consists of *cells* (i.e., elements to which duties must be assigned), grouped into *rows* (i.e., weeks of work) and *columns* (i.e., generic weekdays). The duty types, as specified in the basic schedules, for both rosters in Figure 1 is shown at the top left of the cells, and the numbers indicate the assigned duties. Here, the basic schedule of group B, for example, specifies that the first row starts with a late



Roster Group A



Roster Group B

Figure 1: Example of two rosters for roster group A, consisting of three employees, and B, consisting of four employees. The duty types are indicated by the types (Early, Late, Night, and Rest) above the cells, and the numbers indicate the assigned duties.

duty (L), followed by a day-off (R), and then again a late duty. Similar to Breugem et al. [2017], we assume the basic schedules to be input to the problem.

The crew rostering phase consists of allocating the duties, i.e., days of work, to the basic schedules of the different roster groups. Note that, due the the cyclic nature, also the duties are generic, i.e., the duties are specified on the weekday level. Consider, for example, duty 112, scheduled on Monday in the first row of the roster for group B, as shown in Figure 1. The cyclicity of the roster implies that this duty is executed by the first employee of the group in the first week, by the fourth employee in the second week, and by the third employee in the third week. This process is also known as *rolling out the roster*. Note that the work for an employee of group B repeats itself every four weeks and for an employee of group A every three weeks, since the rosters have has four and three rows, respectively.

The perceived fairness and perceived attractiveness both depend on the way the duties are allocated to the cells of the basic schedules. The perceived fairness relates to the distribution of work among the roster groups (e.g., one roster group should not have much nicer work compared to another roster group), and the perceived attractiveness relates to the structure of these rosters (e.g., sufficient rest time, average workload).

The perceived fairness of the rosters is based on different characteristics of the duties. Since every duty represents a work day, some can be considered more desirable than

others. Duties with many tasks on out-dated rolling stock, for example, are generally considered undesirable. These characteristics are referred to as *duty attributes*. The perceived fairness is measured by the spread (i.e., the difference between the maximal and minimal average value of the duties in the roster) over the groups for the different duty attributes. The smaller this spread, the higher the perceived fairness.

The perceived attractiveness considers the structural characteristics of the rosters, and is defined by so-called *roster constraints*, which forbid or penalize assignments of duties. Well-known examples of roster constraints are rest constraints, enforcing a certain minimum time to rest between consecutive working days, and workload constraints, enforcing a maximum amount of work within a week. The smaller the penalty incurred from the roster constraints, the higher the perceived attractiveness.

Improving both the perceived fairness and perceived attractiveness is not always possible, despite the fact that both metrics aim at improving the quality of the scheduled work. This difference is best illustrated with a simple example, based on the rosters of Figure 1. Suppose the current average duty length is 7 hours and 50 minutes for group A, and 7 hours and 45 minutes for group B. Consider the Monday duties 110 and 112, assigned to groups A and B, respectively. Duty 110 starts at 13:15 and ends at 21:15, hence has a length of 8 hours, and duty 112 starts at 16:30 and ends at 00:10, having a length of 7 hours and 40 minutes. By swapping duties 110 and 112 we would reduce the spread in average duty length, and hence improve the perceived fairness. This swap, however, also implies that the rest period on Tuesday in row B1 becomes shorter, which makes the roster possibly less attractive.

The CCRP-RF can now be stated as follows: Given as input the basic schedules and duties, and a set of fairness levels, determine attractive rosters for each fairness level. That is, determine for each fairness level a roster for each group that minimizes the penalties incurred from the roster constraints, whilst enforcing the desired fairness level.

3 Related Work

Crew planning is a widely studied problem in the literature, dating back as far as Dantzig [1954]. The applications range from health care (e.g., nurse rostering) to transportation (e.g., airline, railway, and bus planning), and the solution methodology covers well-known exact and heuristic methods, such as branch-and-price, simulated annealing, and tabu search. In this section we focus mainly on crew planning within the transportation sector: We refer to Ernst et al. [2004] and Van den Bergh et al. [2013] for more general overviews.

Crew planning is commonly decomposed into two sequential planning phases. In the first phase, known as the crew scheduling or crew pairing problem, the days of work (i.e., duties or pairings) are constructed. This phase mainly focuses on operational cost

(i.e., the number of necessary crew members), together with other key factors, such as the fairness of the work allocation and the constraints resulting from the collective labor agreements. The crew scheduling problem is a well-studied problem in the literature (see, for example, Desrochers and Soumis [1989], Hoffman and Padberg [1993], Grötschel et al. [2003], Abbink et al. [2005], among others), and has been considered in numerous variants, such as rescheduling whenever the underlying tasks are modified (e.g., Lettovskỳ et al. [2000], Potthoff et al. [2010]) and in conjuction with other planning problems, such as aircraft routing and vehicle scheduling (e.g., Cordeau et al. [2001], Huisman et al. [2005a]).

The second planning phase, known as crew rostering, consists of combining the duties (or pairings) into rosters, which are sequences of duties (or pairings) satisfying numerous labor constraints. Typical constraints consider, for example, days off, rest times, variation of work, and personal preferences. Rosters are generally classified as cyclic, i.e., multiple employees working the same roster, and acyclic, i.e., each individual employee working his or her own roster. The latter type is common in the healthcare sector and airline industry (see, for example, Kohl and Karisch [2004] and De Causmaecker and Vanden Berghe [2011], and references therein), whereas the former type is often used in railway operations and mass transit (see Huisman et al. [2005b], Caprara et al. [2007]). Cyclic crew rostering is well-studied in the literature and a variety of formulations have been proposed to model the problem, including a generalized assignment formulation (e.g., Hartog et al. [2009]), a multi-commodity flow formulation (e.g., Caprara et al. [1997], Xie and Suhl [2015], Borndörfer et al. [2015]), and a set covering or set partitioning formulation (e.g., Caprara et al. [1997], Freling et al. [2004], Borndörfer et al. [2015]). The integration of both crew scheduling and rostering has been considered in Mesquita et al. [2013] and Borndörfer et al. [2017], which both propose a solution method based on Benders decomposition. Note that most of aforementioned work focuses on exact methods or heuristics based on mathematical programming techniques, such as column generation. For large-scale and highly complex rostering problems, heuristic methods are sometimes better suited. We refer to Van den Bergh et al. [2013], Table 13, for a detailed overview of solution approaches. In this paper, we build upon the variable-depth neighborhood search heuristic proposed in Borndörfer et al. [2015].

The incorporation of fairness measures in combinatorial optimization problems is, to the best of our knowledge, a relatively young field of research. The fairness of utility allocations, however, has a long history in the economic literature, dating back to the work on bargaining problems by Nash [1950]. Recent work in the field of Operations Research has focused on the trade-off between efficiency (e.g., minimizing cost) and fairness (e.g., maximizing the lowest derived utility). This work includes, among others, the work of Bertsimas et al. [2012] and Bertsimas and Gupta [2015] in the context of air traffic flow management, Bertsimas et al. [2013] on organ allocation, and, in Breugem et al. [2017], on the trade-off between fairness and attractiveness in crew rostering.

The three-phase heuristic extends the exact branch-price-and-cut approach of Breugem et al. [2017] to a solution approach for large-scale instances. It was shown in Breugem et al. [2017] that the exact method is able to solve practically sized instances in reasonable time. For some of the large instances encountered in practice, however, these computation times are considered too high, and quickly found high-quality solutions, although possibly sub-optimal, are preferred. We therefore develop a heuristic taking the exact method as basis, thereby inheriting the strong points of this method, while avoiding excessive computation times. This paper aims at bridging the gap between the exact method developed in Breugem et al. [2017] and the current sequential practice, where first the duties are assigned to the roster groups, and then the rosters per group are optimized separately.

4 Mathematical Formulation

In this section we discuss the mathematical formulation underlying the heuristic. We consider the *row-based* formulation introduced in Breugem et al. [2017] for the FCRP, in which a variable represents the simultaneous assignment of multiple duties to all cells in a row of the basic schedules. The main benefit of this formulation is that all weekly roster constraints, such as weekly variation and maximum workload constraints, can be modeled implicitly in the definition of the variables. In Section 4.1, we introduce the necessary notation and terminology, and in Section 4.2, we present the row-based formulation.

4.1 Notation and Terminology

The row-based formulation models the assignment of duties to the cells by means of *roster sequences*, which specify a simultaneous assignment of duties to a row. In the case of Figure 1, for example, a roster sequence for row A1 has to specify a duty for all three cells (recall that rest days are assumed fixed). In this specific roster, the selected roster sequence specifies the assignment of duty 105 for Wednesday, 111 for Thursday, and 123 for Friday.

Let D denote the set of duties, and let R denote the set of basic schedules. Each basic schedule r is defined by a set of cells T_r, and the set of all cells is denoted by T. An assignment of a duty d to a cell t in a basic schedule will be denoted by the pair (t, d). We define n_r as the total number of duties to be assigned to basic schedule r. Let K denote the set of all rows, and K_r denote the set of rows for basic schedule $r \in R$. We define S_k as the set of all roster sequences for row $k \in K$, where each roster sequence is formally defined as a sequence of assignments (t, d) for the cells in k. The parameter h^k_{ds} indicates whether sequence $s \in S_k$ contains duty d.

The row-based formulation models the assignment of duties to the cells using the decision

variables x_s^k, for all $k \in K$ and $s \in S_k$, indicating whether roster sequence $s \in S_k$ is assigned to row k. The perceived fairness and perceived attractiveness are modeled as follows.

Perceived Fairness

The perceived fairness is expressed in terms of the maximum and minimum average values of different duty attributes (e.g., duty length) among the roster groups. Let A denote the set of duty attributes, and let g_{ad} denote the value of attribute $a \in A$ for duty $d \in D$. Each attribute has a specified lower bound ℓ_a and upper bound u_a, representing the minimum and maximum allowed average values for a basic schedule. In case of the duty length, for example, one could enforce that no roster group works more than 8 hours on average. Besides these bounds, each duty attribute has an associated weight w_a, representing the relative importance of the different duty attributes when calculating the perceived fairness.

The calculation of the perceived fairness is based on the variables v_a and z_a, representing the minimum and maximum average value of duty attribute a among all roster groups, respectively. These variables are linked to the x_s^k variables by means of the constraints

$$\sum_{k \in K_r} \sum_{s \in S_k} \sum_{(t,d) \in s} g_{ad} x_s^k \leq n_r z_a \qquad \forall a \in A, r \in R \tag{1}$$

$$\sum_{k \in K_r} \sum_{s \in S_k} \sum_{(t,d) \in s} g_{ad} x_s^k \geq n_r v_a \qquad \forall a \in A, r \in R, \tag{2}$$

assuring that v_a and z_a attain the minimum and maximum average value for each duty attribute. Given the values v_a and z_a, the fairness level is calculated as

$$\sum_{a \in A} w_a (z_a - v_a), \tag{3}$$

and a fairness budget ζ is enforced by assuring that (3) does not exceed ζ.

Perceived Attractiveness

Each roster constraint penalizes, or forbids, certain combinations of assignments of duties to the cells of the basic schedules. Let P denote the set of roster constraints and let f_{td}^p denote the coefficient for assigning duty d to cell t for roster constraint $p \in P$. For any given assignment of duties to the cells, the violation of the roster constraint is given by the sum of these coefficients minus some allowed threshold value b_p, and this violation is restricted to the interval $\Delta_p = [0, m_p]$. The interval $\Delta_p = [0, 1]$, for example, would allow

a violation of at most one. Each roster constraint $p \in P$ can be written as

$$\sum_{k \in K} \sum_{s \in S_k} \sum_{(t,d) \in s} f_{td}^p x_s^k \leq b_p + \delta_p, \tag{4}$$

where $\delta_p \in \Delta_p$ represents the violation. The perceived attractiveness is maximized by minimizing the sum of roster constraint violations $c_p \delta_p$, where the cost coefficients c_p regulate the relative importance of the different roster constraints. The row-based formulation allows every roster constraint that is *contained* in a row, i.e., it only has non-zero coefficients f_{td}^p for the cells in one row, to be modeled implicitly. Let $P_K \subseteq P$ denote the set of roster constraints that are contained in the rows $k \in K$ (and can therefore be modeled implicitly), and let c_s^k denote the cost associated with sequence $s \in S_k$, that is, c_s^k is the sum of all roster constraint violations in the sequence s.

4.2 Row-Based Formulation

The concepts introduced in Section 4.1 can be integrated to obtain the row-based formulation for the CCRP-FR, similar to the FCRP formulation introduced in Breugem et al. [2017]. For a given fairness budget ζ, the model reads as follows.

$$\min \quad \sum_{k \in K} \sum_{s \in S_k} c_s^k x_s^k + \sum_{p \in P \setminus P_K} c_p \delta_p \tag{5}$$

$$\text{s.t.} \quad \sum_{a \in A} w_a \left(z_a - v_a \right) \leq \zeta \tag{6}$$

$$\sum_{s \in S_k} x_s^k = 1 \qquad \forall k \in K \tag{7}$$

$$\sum_{k \in K} \sum_{s \in S_k} h_{ds}^k x_s^k = 1 \qquad \forall d \in D \tag{8}$$

$$\sum_{k \in K} \sum_{s \in S_k} \sum_{(t,d) \in s} f_{td}^p x_s^k \leq b_p + \delta_p \qquad \forall p \in P \setminus P_K \tag{9}$$

$$\sum_{k \in K_r} \sum_{s \in S_k} \sum_{(t,d) \in s} g_{ad} x_s^k \leq n_r z_a \qquad \forall a \in A, r \in R \tag{10}$$

$$\sum_{k \in K_r} \sum_{s \in S_k} \sum_{(t,d) \in s} g_{ad} x_s^k \geq n_r v_a \qquad \forall a \in A, r \in R \tag{11}$$

$$z_a \leq u_a \qquad \forall a \in A \tag{12}$$

$$v_a \geq \ell_a \qquad \forall a \in A \tag{13}$$

$$x_s^k \in \mathbb{B} \qquad \forall k \in K, s \in S_k \tag{14}$$

$$\delta_p \in \Delta_p \qquad \forall p \in P \setminus P_K \tag{15}$$

$$v_a, z_a \in \mathbb{R} \qquad \forall a \in A. \tag{16}$$

The objective (5) expresses that we minimize the penalties incurred from the roster constraints, partly expressed by the roster sequence costs and partly expressed by the cost of the explicitly modeled roster constraints. The fairness budget is enforced by (6). Constraints (7) and (8) assure that the duties are assigned correctly to the basic schedules: Each row is assigned exactly one roster sequence, and each duty appears in exactly one roster sequence. Constraints (9) model the roster constraint violations, as discussed in Section 4.1. Furthermore, Constraints (10) and (11) assure that the variables v_a and z_a are set to the minimum and maximum value, respectively, while (12) and (13) enforce the lower and upper bounds on the attribute values. Finally, Constraints (14)–(16) express the domains of the decision variables.

5 Three-Phase Heuristic

In this section we present the three-phase heuristic. We first give a general overview of the heuristic and discuss the key components. We then discuss the three different phases separately: In Section 5.1, we discuss the first phase of the heuristic, in which a feasible solution is obtained using a sequential decomposition, and in Sections 5.2, and 5.3, we discuss the paiwise and global improvements phases, respectively.

Recall that for the CCRP-FR the set of fairness levels is considered input. That is, the goal is to determine a solution maximizing attractiveness for a given set of fairness levels, represented by the ordered fairness budgets $\zeta_1 < \ldots < \zeta_n$. In this case, ζ_1 corresponds to the highest fairness level (i.e., smallest budget), and ζ_n to the lowest fairness level (i.e., largest budget). For the sake of explanation, we will often consider the three fairness levels high, medium, and low, but note that the approach does not rely on any assumption regarding the number of fairness levels, nor the size of their corresponding budgets.

The three-phase heuristic aims at quickly finding high-quality solutions for each fairness level. Figure 2 schematically visualizes the algorithm. First, we obtain an initial allocation of the duties to the basic schedules for the highest fairness level, i.e., the lowest fairness budget. This is done by solving a mixed-integer linear program. The roster per basic schedule, given the allocation of the duties, is then optimized using the exact branch-price-and-cut approach developed in Breugem et al. [2017].

The second and third phase of the algorithm both aim at finding profitable re-allocations of duties among the roster groups. In the second phase, we look for pairwise improvements, i.e., we try to find a profitable re-allocation of duties between pairs of roster groups. This is done by solving a reduced problem, based on the linear relaxation of (5)–(16). The key aim of the second phase is to quickly find good solutions for each fairness level, which is achieved by using the found solutions for high levels as starting point for the lower levels.

In the third and final phase of the heuristic we invoke a more time-consuming large-

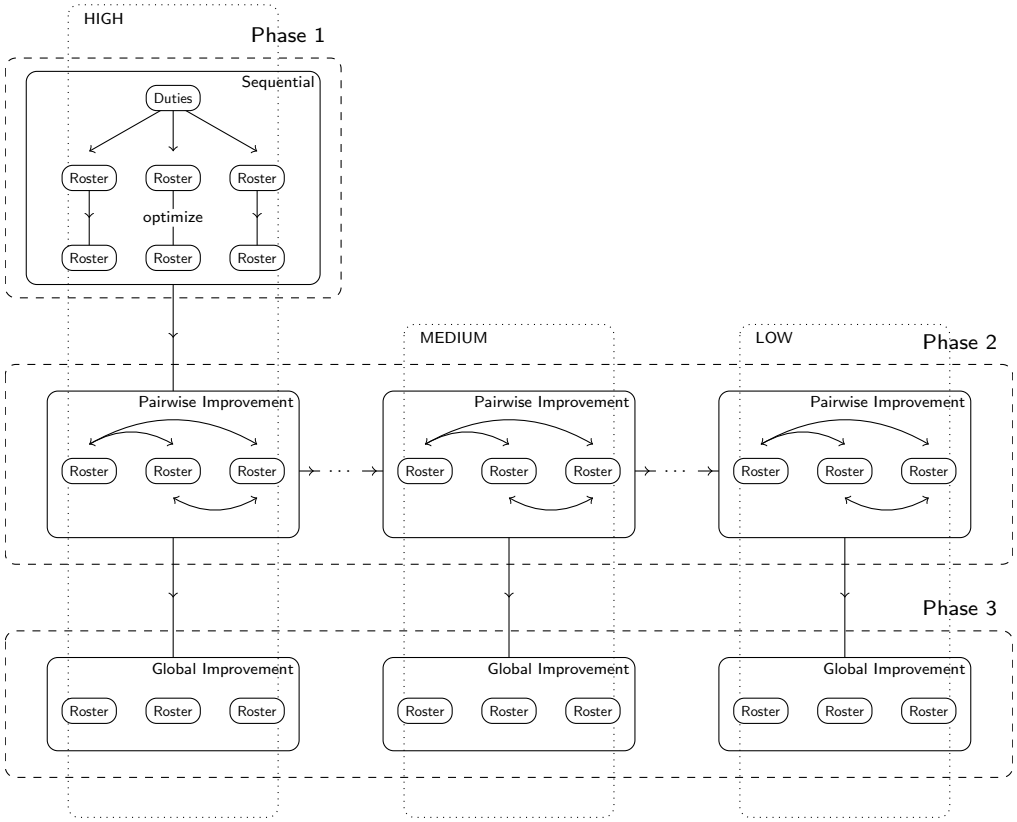


Figure 2: Schematic visualization of the three-phase heuristic. For illustrative purposes, three fairness levels (high, medium, and low) are highlighted. The algorithm starts by finding a feasible solution for the tightest fairness level using a sequential approach (Phase 1). This solution is taken as starting point of a pairwise improvement phase (Phase 2), where we obtain a feasible solution for each fairness level. Finally, the separate solutions are further improved in a global improvement step (Phase 3).

scale neighborhood search algorithm to further improve the solutions. Note that the solutions found in the third phase are not used as starting solutions for the second phase. This assures that, even upon early termination, the algorithm returns a solution for each fairness level, thereby not having to wait for the third, most time consuming, phase to finish. Furthermore, this explicit decoupling would also allow the third phase to be executed in parallel.

5.1 Phase 1: Sequential Decomposition

In the first phase of the algorithm we consider a natural, yet heuristic, decomposition: First, a fair and feasible allocation of the duties to the roster groups is determined, and, secondly, the roster per group is optimized given the allocated duties. This sequential decomposition closely resembles the current practice. The solution found in this phase can therefore be considered as a well-motivated benchmark solution. The allocation of the

duties to the groups is obtained by solving a feasibility problem. We solve this problem using a *cell-based* formulation: Let the binary variable π_{td}, for all $t \in T$ and $d \in D$ indicate whether duty d is assigned to cell t, and let $D_t \subseteq D$ denote the subset of duties compatible with cell t, i.e., those duties for which the weekday and duty type match. Similarly, let $T_d \subseteq T$ denote the cells to which duty d can be assigned. We obtain a feasible allocation by solving the following system of inequalities.

$$\sum_{a \in A} w_a(z_a - v_a) \leq \zeta \tag{17}$$

$$\sum_{d \in D_t} \pi_{td} = 1 \qquad \forall t \in T \tag{18}$$

$$\sum_{t \in T_d} \pi_{td} = 1 \qquad \forall d \in D \tag{19}$$

$$\sum_{t \in T} \sum_{d \in D} f^p_{td} \pi_{td} \leq b_p + m_p \qquad \forall p \in P \tag{20}$$

$$\sum_{t \in T_r} \sum_{d \in D} g_{ad} \pi_{td} \leq n_r z_a \qquad \forall a \in A, r \in R \tag{21}$$

$$\sum_{t \in T_r} \sum_{d \in D} g_{ad} \pi_{td} \geq n_r v_a \qquad \forall a \in A, r \in R \tag{22}$$

$$z_a \leq u_a \qquad \forall a \in A \tag{23}$$

$$v_a \geq \ell_a \qquad \forall a \in A \tag{24}$$

$$\pi_{td} \in \mathbb{B} \qquad \forall t \in T, d \in D \tag{25}$$

$$v_a, z_a \in \mathbb{R} \qquad \forall a \in A. \tag{26}$$

Constraints (18) and (19) assure each cell is assigned exactly one duty and (20) guarantees the feasibility with respect to the roster constraints. Note that, compared to (4), we set δ_p equal to the upper bound m_p. The remaining constraints model the fairness budget, as discussed in Section 4.1. The model (17) – (26) without fairness constraints is similar to the assignment model proposed in Hartog et al. [2009] for cyclic crew rostering.

Given a feasible allocation, the roster for each group is optimized. This sequential decomposition greatly simplifies the problem, as it eliminates the connection between fairness and attractiveness: For a known allocation of duties to the roster groups, the fairness constraints are either satisfied or not, and hence the problem decomposes into a set of (simpler) cyclic crew rostering problems, one for each roster group. For practical roster group sizes, these problems can be solved efficiently using the branch-price-and-cut approach proposed in Breugem et al. [2017].

5.2 Phase 2: Pairwise Improvement

In the second phase of the algorithm we obtain a feasible solution for each fairness budget. First we improve the solution obtained in Phase 1 to obtain a solution for ζ_1, and then

we proceed in an iterative fashion: The solution for the $(i+1)$-th fairness budget ζ_{i+1}, is obtained by searching a neighborhood around the solution for the i-th fairness budget ζ_i, i.e., we exploit the increase of the fairness budget to find profitable re-allocations of duties.

We search for a profitable re-allocation of duties between pairs of roster groups. Note that profitable re-allocations might exist since we increase the fairness budget, hence allocations previously not feasible become feasible. We restrict the re-allocation to pairs of roster groups, to assure that the running time scales well with the instance size. The pairs of roster groups are ordered such that the small groups are considered first. To illustrate this, consider four roster groups A, B, C, and D, of 8, 9, 10, and 12 employees, respectively. We first look for a profitable re-allocation between A and B, then between A and C, then A and D, then between B and C, then B and D, and finally between C and D, each time updating the rosters when a profitable re-allocation has been found. Hence, given k groups, we check $k(k-1)/2$ pairs, after which the improvement step terminates.

The re-allocation is determined by solving the row-based formulation for a reduced set of roster sequences, where the reduction is based on the solution to the linear relaxation: Consider a given feasible integer solution $\hat{x}$ and an optimal (fractional) solution of the linear relaxation $\bar{x}$. For each cell $t \in T$, we allow only a subset $\bar{D}_t \subseteq D_t$ of duties to be assigned. The set $\bar{D}_t$ is initialized by the duty assigned to t in the solution $\hat{x}$, to assure the integer solution remains feasible, and is then enriched based on $\bar{x}$: For each $k \in K$ and $t \in k$, we enlarge $\bar{D}_t$ by adding the duties $d \in D_t$ for which

$$\sum_{\substack{s \in S_k: \\ s \ni (t,d)}} \bar{x}_s^k > 0,$$

i.e., we add those duties to $\bar{D}_t$ that have a non-zero coefficient for t in $\bar{x}$. Figure 3 illustrates this reduction. Generally, not too many duties are selected this way.

The reduced set of roster sequences $\bar{S}_k \subseteq S_k$ for each row $k \in K$, consists of exactly those roster sequences assigning only duties in $\bar{D}_t$ to each $t \in k$. This set is determined by complete enumeration. The resulting reduced model is solved using a commercial solver to obtain a possible improved allocation of the duties. In this phase of the heuristic, we also solve the linear relaxation of (5)– (16) for all roster groups simultaneously to obtain a lower bound on the optimal solution value.

5.3 Phase 3: Global Improvement

In the third and final step of the algorithm, each of the solutions is further improved using a sophisticated local search algorithm. In this phase of the heuristic we aim at finding improving duty exchanges between *all* roster groups (hence, the name *global improvement*),

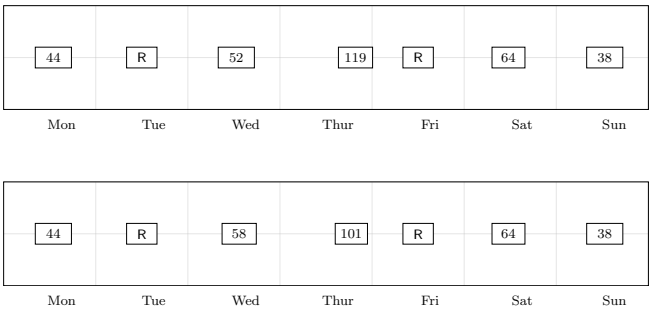


Figure 3: An example of the problem reduction. Suppose the shown roster sequences are assigned positive values in $\bar{x}$ for some given row. The allowed duties obtained from $\bar{x}$ are given by $\bar{D}_{\text{Mon}} = \{44\}$, $\bar{D}_{\text{Wed}} = \{52, 58\}$, $\bar{D}_{\text{Thur}} = \{101, 119\}$, $\bar{D}_{\text{Sat}} = \{64\}$, and $\bar{D}_{\text{Sun}} = \{38\}$. In this example, two additional roster sequences will be generated leading to a total of four roster sequences.

as opposed to Phase 2, where we only consider pairs of roster groups. This is done using local search.

The proposed local search algorithm is based on variable-depth neighborhood search (VDNS), a neighborhood search technique belonging to the family of so-called very large-scale neighborhood search algorithms (see e.g., Ahuja et al. [2002], Pisinger and Ropke [2010] for a detailed overview). The key idea behind VDNS is to (heuristically) explore a large neighborhood by constructing a *chain* of moves, with each move belonging to a smaller neighborhood. In this way, a large part of the solution space is searched, including moves involving a large number of elements, whilst avoiding excessive computation times. This simple yet successful idea dates back to Lin and Kernighan [1973], who used it to construct an efficient heuristic for the Traveling Salesman Problem (TSP). The concept of VNDS has been applied to a wide range of difficult combinatorial optimization problems ever since.

To apply VDNS one first needs to define a parametrized neighborhood N_k, satisfying

$$N_1 \subseteq \ldots \subseteq N_i \subseteq \ldots \subseteq N_n.$$

Typical examples of such neighborhoods are the k-permutation neighorhood, i.e., permuting k elements of the solution, or the k-arc exchange neighborhood considered in Lin and Kernighan [1973]. The key idea is to first pick a suitably sized neighborhood N_i, large enough to escape local optima and small enough to be searched efficiently, and heuristically explore N_n by chaining together moves in the smaller neighborhood N_i. The chain is constructed by making a profitable move in N_i, fixing the involved elements, and repeating this until no more profitable moves exist. As noted in Johnson and McGeoch [1997], this chaining of moves can be seen as a special type of tabu search, using a flexibly sized tabu list. Figure 4 gives a schematic visualization of VDNS.

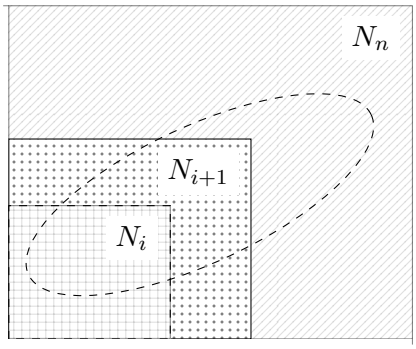
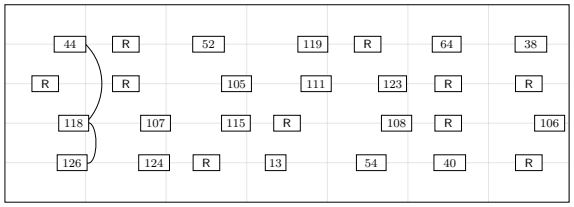


Figure 4: Schematic visualization of VDNS. The parametrized neighborhood N_k is heuristically searched (indicated by the dashed area) by chaining N_i moves. The resulting move can be part of N_i, N_{i+1}, or even N_n, implying that the 'depth' of the move is not fixed.

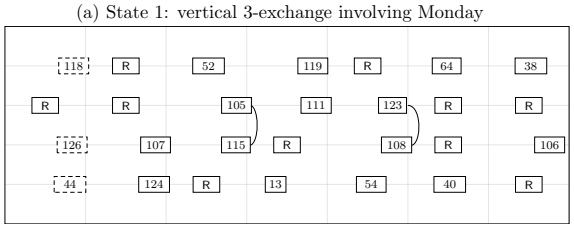
Crucial for VDNS to work is a suitably picked neighborhood. In particular, the neighborhood should (*i*) be able to escape local optima, (*ii*) be searchable in reasonable time, and (*iii*) guarantee that the solution remains feasible, or at least can be easily made feasible. The latter depends heavily on the structure of the underlying problem. In case of a fixed basic schedule, for example, duties cannot be freely exchanged (e.g., a late duty on Monday cannot be exchanged with an early duty on Monday nor a late duty on Tuesday). We therefore propose two neighborhoods for the VDNS algorithm, based on two different duty exchange operations: *vertical* and *horizontal* duty exchanges.

Vertical k-exchanges are exchanges between k duties of the same type and in the same column, i.e., they are vertically aligned. This assures that the involved duties can always be exchanged without violating the basic schedule, i.e., the structure of the solution is not affected by a vertical exchange. The feasibility with respect to the roster constraints can be readily checked when performing an exchange, hence the feasibility of the solution can always be assured. Note that the vertical k-exchange neighborhood can be searched in $\mathcal{O}\left(|D|^k\right)$ time. Figure 5a gives an example of a vertical 3-exchange. We will denote the vertical k-exchange neighborhood by V_k.

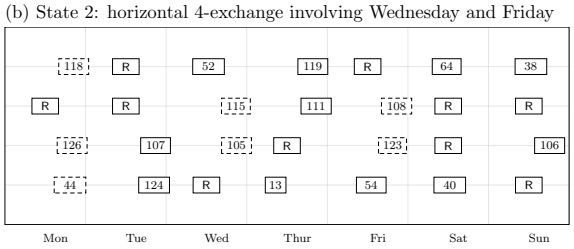
Horizontal k-exchanges are a more involved type of exchange, affecting duties of different types and in different columns. The horizontal exchange neighborhood aims at complementing the vertical exchange neighborhood, which can get stuck in local optima due to the restriction to one single column. Formally, a horizontal k-exchange, for k even, is a sequence of $k/2$ vertical 2-exchanges, where each 2-exchange shares at least one row with its predecessor. Figure 5b gives an example of a horizontal 4-exchange. By considering a sequence of multiple vertical 2-exchanges we allow duties of different weekdays and types to be affected within a single exchange, i.e., one exchange can involve multiple duties in one row. Furthermore, we limit the search time by only considering sequences where consecutive exchanges share a row: It is not difficult to show that the horizon-



(a) State 1: vertical 3-exchange involving Monday



(b) State 2: horizontal 4-exchange involving Wednesday and Friday



(c) State 3: final roster

Figure 5: Example of horizontal and vertical exchanges. First, a vertical 3-exchange is performed on the the Monday duties 44, 118 and 126. Then, a horizontal 4-exchange is performed on the Wednesday duties 105 and 115, and the Friday duties 108 and 123.

tal k-exchange neighborhood can be searched in $\mathcal{O}\left(|D|^{k/2+1}\right)$ time. We will denote the horizontal k-exchange neighborhood by H_k.

The proposed VDNS algorithm combines chains of horizontal 4- and vertical 3-exchanges, searchable in $\mathcal{O}\left(|D|^3\right)$ time, with horizontal 6- and vertical 4-exchanges, searchable in $\mathcal{O}\left(|D|^4\right)$ time. This is done similar to the Dynamic Depth-EXchange (DEX) algorithm, introduced in Borndörfer et al. [2015]: We combine H_4+V_3 chains, i.e., chains of horizontal 4- and vertical 3-exchanges, with single $H_6 + V_4$ moves to escape local optima. These neighborhoods are large enough to escape local optima, and small enough to be considered efficient.

The final VDNS algorithm is shown in Figure 6. Starting from an initial solution, we construct improving $H_4 + V_3$ chains. Here we allow for some small deterioration in the chaining process, to add more flexibility to the search. Once the chaining procedure finishes, i.e., the best $H_4 + V_3$ move among the non-fixed duties leads to too much of a cost increase, the solution is updated and all duties are removed from the set of fixed duties. If an improving chain was found, we again search for an improving chain for the updated solution. Otherwise, we try to escape the current local optimum by using one (strictly profitable) $H_6 + V_4$ move. If this succeeds, we repeat the chaining procedure for the updated solution. Otherwise the algorithm terminates and the final solution is found.

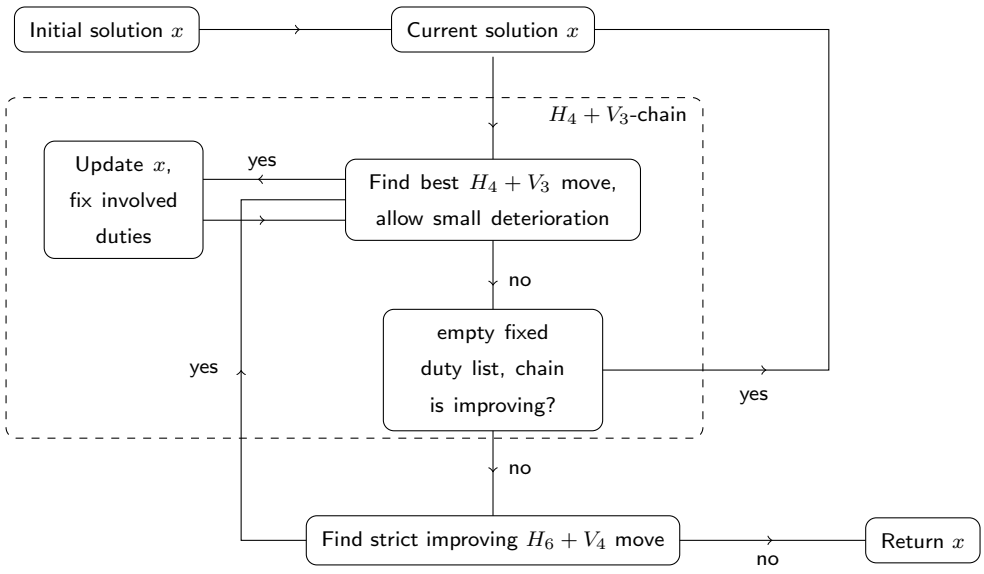


Figure 6: Schematic visualization of the VDNS algorithm. An initial solution is improved using $H_4 + V_3$ chains. If no improving chain can be found, a single $H_6 + V_4$ move is considered. In case no profitable $H_6 + V_4$ move exists, the algorithm terminates. Otherwise, the algorithm continues the search for improvements using $H_4 + V_3$ chains.

6 Computational Experiments

To evaluate the performance of the heuristic, we apply our solution approach to different instances based on data from NS. In Section 6.1 we discuss the experimental set-up and in Section 6.2 we show the computational results.

6.1 Experimental Set-Up

We consider a total of 10 instances. The first four instances are those considered in Breugem et al. [2017], and can be used to validate the performance of the heuristic, as the optimal solutions are known for these instances. For the remaining six (larger) instances, no optimal solutions are known.

The instances each consist of basic schedules and duties that need to be assigned to these schedules. The basic schedules specify a type, i.e., a duty type or a day off, for each cell. The considered duty types are early, late, and night. Based on the types of the duties that need to be assigned, each roster group can be (roughly) categorized in one of three categories: early, late-night, and mixed (i.e., all three types). We refer to these categories as E, LN, and M, respectively.

For each instance, Table 1 shows the number of duties of each type, and the size of the groups per category (i.e., the entry 12/8 for LN means the instance contains one LN group

	Duties				Employees			
	E	L	N	Total	E	LN	M	Nr Groups
1	55	29	29	113	14	12	4	3
2	58	33	24	115	12	12	4	3
3	38	22	11	71	6	6	6	3
4	37	17	15	69	6	8	6	3
5	74	62	55	191	12	12/12	12	4
6	88	36	31	155	14	8	12/6	4
7	86	36	37	159	12/6	12	12	4
8	74	30	16	120	6/6	8	12	4
9	126	70	54	250	14/6	12/8	12/12	6
10	142	63	61	266	12/12	12/8	12/12	6

Table 1: Characteristics of the instances. For each instance, the number of duties of type early (E), late (L), and night (N) is shown, together with the total number of duties, and the number of employees of each category early (E), late-night (LN), and mixed (M). Finally, the number of roster groups per instance is shown.

of 12 employees and one LN group of 8 employees). The first four instances all consist of three groups, one of each type, of varying sizes. The second four instances all consist of four groups of varying type and varying size. The fifth instance is the largest among these four, with approximately 190 duties, whereas the eighth instance is the smallest, with roughly 120 duties. The final two instances both consist of six groups, with two groups of each type. The total number of duties for both instances is roughly 250. The size of these instances corresponds to the size of the pilot study mentioned in Section 1.

The perceived fairness is based on five different duty attributes. These include three attributes concerning the scheduled work: the percentages of *high quality work* (e.g., Intercity work), *aggression work* (e.g., trips where passengers are less likely to have a ticket, and hence might become aggressive), and *double decker work* (as work on double decker trains is physically more demanding). Furthermore, there are two attributes regarding the entire duty: the *duty length*, and the *repetition within duty* (as duties with many of the same trips are repetitive, which is considered undesirable).

The perceived attractiveness is determined by four types of roster constraints. These types can be divided into two classes. The first class contains 'binary' roster constraints, i.e., roster constraints linking exactly two cells in the basic schedule. This class contains the so-called *rest time* end *rest day* constraints: It is required that an employee has a minimum rest time after each duty. After a night duty this rest time should be 14 hours and after any other type of duty it should be 12 hours. Furthermore, rest times shorter than 16 hours are penalized. In addition, the length of each scheduled rest period has to be sufficient. This implies that there is a minimal time enforced between duties scheduled before and after rest days. The enforced rest time is 6 hours plus 24 hours

for each rest day. The second class of roster constraints consists of 'row-based' roster constraints. This class contains *workload* constraints, i.e., the total workload in a row is not allowed to exceed 45 hours, and a large collection of *variation* constraints. This latter type of constraint aims at balancing different duty attributes (e.g., duty length, percentage double decker work) equally over the rows, which is achieved by penalizing positive deviations from the average (measured over all duties) for each row in the roster. In total we consider variation constraints for 9 different attributes. Note that the stronger bound obtained from the row-based formulaton (compared to the cell-based formulation) results from capturing the row-based roster constraints directly in the roster sequence costs.

6.2 Computational Results

We evaluated the heuristic solution method on each of the ten instances using different fairness budgets. The cell-based formulation used in the first phase is solved using CPLEX 12.7.1 (from hereon simply referred to as CPLEX), with a time limit of 30 minutes, i.e., whenever no feasible allocation is found within 30 minutes the fairness budget is considered infeasible. The optimization problem per roster group, when a feasible allocation is found, is solved to optimality using the branch-price-and-cut approach developed in Breugem et al. [2017]. For the pairwise improvement step in the second phase, we allow a solving time of two minutes per pair. The linear relaxation of the row-based formulation is solved using the column generation approach proposed in Breugem et al. [2017], and the reduced problem is again solved using CPLEX.

The results for the first four instances are shown in Table 2. For each instance, we consider five different fairness levels: extreme, high, moderate, low, and poor. The corresponding fairness budgets are obtained as follows. We first determine the exact trade-off curve between perceived fairness and attractiveness, using the branch-price-and-cut algorithm of Breugem et al. [2017]. We then pick the fairness budgets for each level based on the quantiles of the trade-off curve: extreme corresponds to the leftmost (i.e., fairest) point, high to the 10% quantile, moderate to the 25% quantile, low to the 50% quantile, and, finally, poor to the 75% quantile of the curve. In this way, we assure that the fairness levels cover the entire spectrum of possible solutions.

Table 2 shows that the heuristic finds high-quality, close to optimal, solutions for all but a few instances. For the extreme fairness level, the solution found in the first phase coincides with the optimal solution for all four instances. As Table 2 shows, the major improvement is often achieved in the second phase of the algorithm, i.e., the pairwise improvement phase. The third phase generally improves only slightly upon this solution. There are, however, also some cases (e.g., instance 2) in which the third phase greatly improves upon the second phase. Note that for most of the instances, there is a substantial integrality gap.

	Fairness	Phase 1	Phase 2	Phase 3	Optimal	Root	Gain (%)	Gap (%)
	Extreme	1175.8	1175.8	1175.8	1175.8	1155.6	0.0	0.0
	High		1172.6	1172.6	1169.2	1133.4	0.3	0.3
1	Moderate		1172.3	1172.3	1164.2	1128.5	0.3	0.7
	Low		1172.3	1165.8	1133.8	1115.4	0.8	2.7
	Poor		1143.1	1142.7	1120.8	1104.8	2.8	1.9
	Extreme	1288.0	1288.0	1288.0	1288.0	1172.3	0.0	0.0
	High		1288.0	1248.6	1216.4	1158.3	3.1	2.6
2	Moderate		1257.3	1226.3	1192.2	1149.3	4.8	2.8
	Low		1257.3	1210.2	1186.1	1135.8	6.0	2.0
	Poor		1204.5	1196.7	1177.8	1126.6	7.1	1.6
	Extreme	979.3	979.3	979.3	979.3	800.9	0.0	0.0
	High		979.3	979.3	853.8	793.4	0.0	12.8
3	Moderate		859.0	859.0	830.3	788.4	12.3	3.3
	Low		802.4	802.4	794.0	781.5	18.1	1.0
	Poor		791.3	787.5	787.5	777.6	19.6	0.0
	Extreme	910.5	910.5	910.5	910.5	776.7	0.0	0.0
	High		872.1	872.1	847.2	767.3	4.2	2.9
4	Moderate		872.1	872.1	827.7	761.3	4.2	5.1
	Low		808.4	808.4	788.0	746.8	11.2	2.5
	Poor		774.2	774.2	774.2	742.7	15.0	0.0

Table 2: Results for the first four instances for five different fairness levels. For each instance and each fairness level, we show the objective values, i.e., perceived attractiveness penalty, obtained in the different phases of the heuristic, the optimal solution value, the root bound, the gain compared to the benchmark solution (i.e., Phase 1), and the gap with the optimal solution.

The computation times for each instance and each fairness level are shown in Table 3. The overall computation times are decomposed per fairness level and phase of the algorithm. Furthermore, we show the total computation times for the solutions for each fairness level found in Phases 2 and 3: The sequential nature of Phase 2 implies that solutions for low fairness levels are found only after the solutions for the higher fairness levels are already computed. The total computation times take these additional computations into account.

Table 3 shows that in almost all cases the computation time of the heuristic is an order of magnitude smaller than the time necessary for the exact approach. Only for a few fairness levels, the computation time of the branch-price-and-cut algorithm is comparable with those of the heuristic. For instances 3 and 4, we observe that the heuristic only needs half a minute, wheres the exact approach can take up to more than one hour. Finally, Table 3 shows that the time necessary for the pairwise improvement step in Phase 2, is substantially smaller than the time necessary for Phase 3.

For instances 5 to 10, computing the exact trade-off curve is computationally intractable. We therefore solve each instance for four fairness levels, and compare with the lower bound based on the linear relaxation of the row-based formulation. We consider the fairness levels extreme, high, moderate, and low. These fairness levels correspond to the a priori determined fairness budgets 2, 3, 5 and 10. To intuitively understand and relate

	Fairness	Phase 1	Phase 2	Total Phase 2	Phase 3	Total Phase 3	Optimal
	Extreme	21	25	46	171	217	99
	High		13	59	172	231	979
1	Moderate		28	87	168	255	1102
	Low		25	112	191	303	507
	Poor		15	127	193	320	1009
	Extreme	83	77	160	132	292	479
	High		77	237	172	409	460
2	Moderate		76	313	174	487	533
	Low		77	390	163	553	721
	Poor		35	425	172	597	2769
	Extreme	0	1	1	24	25	2315
	High		1	2	24	26	1093
3	Moderate		0	2	26	28	686
	Low		0	2	25	27	50
	Poor		0	2	24	26	28
	Extreme	2	3	5	23	28	5346
	High		3	8	23	31	2498
4	Moderate		1	9	23	32	5044
	Low		0	9	25	34	1306
	Poor		0	9	25	34	629

Table 3: Computation times for the first four instances for five different fairness levels. For each instance and each fairness level, the computation times for each phase are shown (in seconds). Furthermore, we show the total computation time for Phase 2 and 3, since Phase 2 is solved sequentially, and hence the computation times are cumulative. Finally, the computation times of the branch-price-and-cut algorithm are shown.

the chosen fairness budgets, consider that, for groups of about 10 employees, a two unit difference in fairness budget could imply that one group is assigned one full Intercity duty less than the other groups, or has to work about half an hour longer in one row of the schedule compared to the other groups. From a practical point of view, such differences are substantial and undesirable, also because these differences persist throughout the entire year. The results are shown in Table 4.

Table 4 shows that the heuristic increases the attractiveness greatly: For 12 out of 18 instances, the improvement was more than 20%, and only for one instance the improvement was not substantial. The improvement for the largest two instances is especially large. Furthermore, the largest gains are obtained for the less strict fairness levels. This is intuitive, as the focus on fairness is less, and hence the loss of not taking attractiveness into account when allocating the duties over the groups will be larger. Note that a major part of the improvement is in the pairwise improvement phase, i.e., the second phase, and only a small further improvement is obtained in the third phase. For some instances, however, the third phase allows for a substantial improvement (e.g., for instance 8). The gap with the root bound indicates that the found solutions are of high quality, although in some cases a substantial gap is still present. We note, however, that a substantial integrality gap can be expected, especially for tight fairness budgets, as was also noted in Table 2.

	Fairness	Phase 1	Phase 2	Phase 3	Root	Gain (%)	Gap (%)
5	Extreme	2041.5	2041.5	2039.5	1542.5	0.1	24.4
	High		1780.2	1778.2	1529.0	12.9	14.0
	Moderate		1668.0	1630.5	1514.5	20.1	7.1
	Low		1546.2	1544.1	1491.8	24.4	3.4
6	Extreme	1811.7	1651.6	1650.4	1296.4	8.9	21.4
	High		1513.2	1512.0	1291.5	16.5	14.6
	Moderate		1412.4	1411.1	1282.3	22.1	9.1
	Low		1358.2	1356.9	1265.7	25.1	6.7
7	Extreme	1956.3	1559.0	1557.8	1328.4	20.4	14.7
	High		1510.4	1503.7	1318.0	23.1	12.3
	Moderate		1469.5	1465.1	1299.2	25.1	11.3
	Low		1412.1	1410.7	1262.5	27.9	10.5
8	Moderate	1502.0	1397.4	1396.9	1120.4	7.0	19.8
	Low		1292.7	1207.9	1097.8	19.6	9.1
9	Moderate	2877.0	2252.1	2249.2	1774.0	21.8	21.1
	Low		2194.0	2182.3	1768.2	24.1	19.0
10	Moderate	3009.0	1863.7	1859.2	1526.0	38.2	17.9
	Low		1759.6	1759.6	1486.5	41.5	15.5

Table 4: Results for instances 5 to 10. Each instance is solved for four different fairness levels: extreme, high, moderate, and low. For each instance and each fairness level, we show the objectives obtained in the different phases of the heuristic, the gain compared to the benchmark solution (i.e., Phase 1), and the gap with the root bound obtained from the row-based formulation. Omitted rows indicate that no feasible allocation could be found within the time limit of 30 minutes.

The computation times for the second set of instances are shown in Table 5. Similar to Table 3, we show the computation times per fairness level and phase of the algorithm, together with the total computation times for Phases 2 and 3.

The running times shown in Table 5 can be considered well-suited for practice: The second phase has a running time of a few minutes, the same order of magnitude as the first phase, and the third phase stays within two hours for all instances. Note that, for instances 8, 9, and 10, the running time for Phase 1 does not include the time for fairness levels Extreme and High, for which no solution could be found within the time limit of half an hour.

Summarizing, our experiments show that the heuristic approach complements the exact branch-and-price method developed in Breugem et al. [2017]. The algorithm is able to find close-to-optimal solutions for the first set of instances, and greatly improves upon the sequential approach for the second set of instances. Furthermore, a major part of this improvement is due to the pairwise improvement in the second phase of the algorithm, which runs orders of magnitude faster than the branch-and-price approach. The third phase is more time consuming, but can realize a substantial improvement.

	Fairness	Phase 1	Phase 2	Total Phase 2	Phase 3	Total Phase 3
5	Extreme	167	301	468	1020	1488
	High		214	682	1079	1761
	Moderate		25	707	1232	1939
	Low		15	722	1068	1790
6	Extreme	56	201	257	731	988
	High		47	304	767	1071
	Moderate		11	315	740	1055
	Low		8	323	815	1138
7	Extreme	512	92	604	695	1299
	High		11	615	728	1343
	Moderate		30	645	871	1516
	Low		8	653	746	1399
8	Moderate	7	46	53	309	362
	Low		4	57	361	418
9	Moderate	23	328	351	4594	4945
	Low		26	377	5452	5829
10	Moderate	127	58	185	6335	6520
	Low		38	223	5546	5769

Table 5: Computation times for the instances 5 to 10, for the four different fairness levels. For each instance and each fairness level, the computation times for each phase are shown (in seconds). Furthermore, we show the total computation time for the solutions found in Phase 2 and 3, to account for the sequential approach used in Phase 2. Omitted rows indicate that no feasible allocation could be found within the time limit of 30 minutes.

7 Conclusion

In this paper, we proposed a heuristic method for the Cyclic Crew Rostering Problem with Fairness Requirements (CCRP-FR), a variant of the Fairness-oriented Crew Rostering Problem (FCRP). In this problem, attractive rosters have to be constructed for a fixed, a priori known, set of fairness levels. The development of the heuristic solution approach is motivated by practice: The crew rostering problem is generally solved multiple times for varying parameter settings, implying that, even during the tactical planning phase, it is desirable that high quality solutions can be obtained quickly. Also, the underlying complexity of the problem implies that exact methods are incapable of coping with some of the large instances encountered in practice.

The developed three-phase heuristic combines the strengths of the exact apprach for the FCRP developed in Breugem et al. [2017] with a large-scale neighborhood search algorithm. The design of the heuristic assures that good solutions for all fairness levels are obtained quickly, and can still be further improved if additional running time is available.

We evaluated the heuristic on real-world data from NS. We showed that the heuristic finds close to optimal solutions for many of the considered instances. Furthermore, the

computation times are generally an order of magnitude smaller than the time necessary for the branch-price-and-cut approach. In particular, the computational results show that the heuristic is able to quickly find major improvements upon the sequential approach: For most instances the heuristic is able to reduce the attractiveness penalty by at least 20% in just a few minutes. Furthermore, the heuristic also provides a lower bound which gives an estimate of the solution quality. This bound indicates that the heuristic finds high-quality solutions, also for the instances for which no optimal solution is known. The running time of the heuristic can be considered well-suited for practice, even for the largest instances: The second phase has a running time of a few minutes, similar to the first phase, and the third phase stays within two hours for all instances.

References

E. Abbink, M. Fischetti, L. Kroon, G. Timmer, and M. Vromans. Reinventing crew scheduling at Netherlands Railways. *Interfaces*, 35(5):393–401, 2005.

R. K. Ahuja, Ö. Ergun, J. B. Orlin, and A. P. Punnen. A survey of very large-scale neighborhood search techniques. *Discrete Applied Mathematics*, 123(1-3):75–102, 2002.

D. Bertsimas and S. Gupta. Fairness and collaboration in network air traffic flow management: an optimization approach. *Transportation Science*, 50(1):57–76, 2015.

D. Bertsimas, V. F. Farias, and N. Trichakis. On the efficiency-fairness trade-off. *Management Science*, 58(12):2234–2250, 2012.

D. Bertsimas, V. F. Farias, and N. Trichakis. Fairness, efficiency, and flexibility in organ allocation for kidney transplantation. *Operations Research*, 61(1):73–87, 2013.

R. Borndörfer, M. Reuther, T. Schlechte, C. Schulz, E. Swarat, and S. Weider. Duty rostering in public transport-facing preferences, fairness, and fatigue. In *CASPT*, 2015.

R. Borndörfer, C. Schulz, S. Seidl, and S. Weider. Integration of duty scheduling and rostering to increase driver satisfaction. *Public Transport*, 9(1-2):177–191, 2017.

T. Breugem, T. Dollevoet, and D. Huisman. Is equality always desirable? Analyzing the trade-off between fairness and attractiveness in crew rostering. Technical Report EI2017-30, Econometric Institute, 2017.

A. Caprara, M. Fischetti, P. Toth, D. Vigo, and P. L. Guida. Algorithms for railway crew management. *Mathematical Programming*, 79(1-3):125–141, 1997.

A. Caprara, L. Kroon, M. Monaci, M. Peeters, and P. Toth. Passenger railway optimization. *Handbooks in Operations Research and Management Science*, 14:129–187, 2007.

J. F. Cordeau, G. Stojković, F. Soumis, and J. Desrosiers. Benders decomposiion for simultaneous aircraft routing and crew scheduling. *Transportation science*, 35(4):375–388, 2001.

G. B. Dantzig. 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, 1954.

P. De Causmaecker and G. Vanden Berghe. A categorisation of nurse rostering problems. *Journal of Scheduling*, 14(1):3–16, 2011.

M. Desrochers and F. Soumis. A column generation approach to the urban transit crew scheduling problem. *Transportation Science*, 23(1):1–13, 1989.

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

R. Freling, R. M. Lentink, and A. P. M. Wagelmans. A decision support system for crew planning in passenger transportation using a flexible branch-and-price algorithm. *Annals of Operations Research*, 127(1-4):203–222, 2004.

M. Grötschel, R. Borndörfer, and A. Löbel. Duty scheduling in public transit. In *Mathematics-Key Technology for the Future*, pages 653–674. Springer, 2003.

A. Hartog, D. Huisman, E. Abbink, and L. Kroon. Decision support for crew rostering at NS. *Public Transport*, 1(2):121–133, 2009.

K. L. Hoffman and M. Padberg. Solving airline crew scheduling problems by branch-and-cut. *Management Science*, 39(6):657–682, 1993.

D. Huisman, R. Freling, and A. P. M. Wagelmans. Multiple-depot integrated vehicle and crew scheduling. *Transportation Science*, 39(4):491–502, 2005a.

D. Huisman, L. G. Kroon, R. M. Lentink, and M. J. C. M. Vromans. Operations research in passenger railway transportation. *Statistica Neerlandica*, 59(4):467–497, 2005b.

D. S. Johnson and L. A. McGeoch. The traveling salesman problem: A case study in local optimization. *Local search in combinatorial optimization*, 1(1):215–310, 1997.

N. Kohl and S. E. Karisch. Airline crew rostering: Problem types, modeling, and optimization. *Annals of Operations Research*, 127(1-4):223–257, 2004.

L. Lettovskỳ, E. L. Johnson, and G. L. Nemhauser. Airline crew recovery. *Transportation Science*, 34(4):337–348, 2000.

S. Lin and B. W. Kernighan. An effective heuristic algorithm for the traveling-salesman problem. *Operations Research*, 21(2):498–516, 1973.

M. Mesquita, M. Moz, A. Paias, and M. Pato. A decomposition approach for the integrated vehicle-crew-roster problem with days-off pattern. *European Journal of Operational Research*, 229(2):318–331, 2013.

J. Nash. The bargaining problem. *Econometrica: Journal of the Econometric Society*, pages 155–162, 1950.

D. Pisinger and S. Ropke. Large neighborhood search. In *Handbook of metaheuristics*, pages 399–419. Springer, 2010.

D. Potthoff, D. Huisman, and G. Desaulniers. Column generation with dynamic duty selection for railway crew rescheduling. *Transportation Science*, 44(4):493–505, 2010.

J. Van den Bergh, J. Beliën, P. De Bruecker, E. Demeulemeester, and L. De Boeck. Personnel scheduling: A literature review. *European Journal of Operational Research*, 226(3):367–385, 2013.

L. Xie and L. Suhl. Cyclic and non-cyclic crew rostering problems in public bus transit. *OR Spectrum*, 37(1):99–136, 2015.