Analyzing a Family of Formulations for Cyclic Crew Rostering

Thomas Breugem[1]
Technology and Operations Management, INSEAD, Fontainebleau, France
thomas.breugem@insead.edu

Twan Dollevoet
Erasmus University Rotterdam, The Netherlands
dollevoet@ese.eur.nl

Dennis Huisman
Erasmus University Rotterdam, The Netherlands
Netherlands Railways, The Netherlands
huisman@ese.eur.nl

Abstract

In this paper, we analyze a family of formulations for the Cyclic Crew Rostering Problem (CCRP), in which a cyclic roster has to be constructed for a group of employees. Each formulation in the family is based on a partition of the roster. Intuitively, finer partitions give rise to a formulation with fewer variables, but possibly more constraints. Coarser partitions lead to more variables, but might allow to incorporate many of the constraints implicitly. We derive analytical results regarding the relative strength of the different formulations, which can serve as a guideline for formulating a given problem instance. Furthermore, we propose a column generation approach, and use it to compare the strength of the formulations empirically. Both the theoretical and computational results demonstrate the importance of choosing a suitable formulation. In particular, for practical instances of Netherlands Railways, stronger lower bounds are obtained, and more than 90% of the roster constraints can be modeled implicitly.

2012 ACM Subject Classification Applied computing → Transportation

Keywords and phrases Crew Planning, Roster Sequence, Column Generation, Railway Optimization

Digital Object Identifier 10.4230/OASIcs.ATMOS.2020.7

1 Introduction

The construction of rosters (often referred to as crew rostering) is an important part of the planning process at a public transport operator. As opposed to many other planning problems at a railway operator (e.g., rolling stock scheduling), the main goal in crew rostering is not to minimize costs. Instead the goal is to maximize the attractiveness of the rosters from the point of view of the employees. This implies that, for example, the rest time between consecutive working days and the variation of work over a week have to be taken into account when constructing the rosters. Altogether, this leads to a complex optimization problem.

The inclusion of attractiveness in crew planning has shown to be important in practice. In the Netherlands, for example, the incorporation of attractiveness in crew planning was vital in resolving conflicts between the labor unions and Netherlands Railways (NS), the largest railway operator in the Netherlands. An import development in this respect was the introduction of the "Sharing-Sweet-and-Sour" rules, which aim to increase the quality of

[1] The majority of this work was done while working at Erasmus University Rotterdam, The Netherlands.

20th Symposium on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2020).
Editors: Dennis Huisman and Christos D. Zaroliagis; Article No. 7; pp. 7:1–7:16

OpenAccess Series in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

work (see [1] for a detailed discussion). These rules, for example, assure that "nice work" is equally distributed among all depots. Similarly, [3] discusses the importance of attractive work in Germany.

The focus on a fair distribution of work is apparent in the use of roster groups, which are groups of employees with similar characteristics (e.g., age, preferences), that operate in cyclic rosters. This implies that, after a certain time period, each employee in a group has done the exact same work, assuring a fair distribution of work within each group. In the Cyclic Crew Rostering Problem (CCRP) the goal is to maximize the attractiveness of a roster constructed for such a group.

Various models for the CCRP have been developed in the scientific literature. These models generally belong to one of three categories: generalized assignment, multi-commodity flow, and set partitioning models. [6] and [2] consider both a multi-commodity flow model and a set partitioning model for crew rostering. Furthermore, both argue which formulation is more suitable, given the constraint set: [6] stresses that flow-based formulations are well-suited for problems where the main focus is on the follow-up of duties, whereas a set partitioning formulation is better suited for problems where the feasibility and cost depend on the overall duty sequence. Similarly, [2] notes that the set partitioning formulation is preferred when many difficult roster constraints have to be taken into account. [12] and [9] propose an assignment model with side constraints. They solve the problem using a two-phase decomposition, in which first the "skeleton" of the roster is optimized (e.g., days-off are determined), and then the duties are assigned. [13] proposes a multi-commodity flow formulation for both cyclic and acyclic crew rostering, and applies both models to practical instances from a German bus company. Finally, [11] considers both assignment and multi-commodity flow models for the bus driver rostering problem with day-off patterns, and provides theoretical results regarding the relative strength of the models.

In this paper, we provide an in-depth analysis of modeling techniques for the CCRP. We propose a family of formulations, and derive analytical results regarding the relative strength of the proposed formulations. The family of formulations can be seen as a generalization of the typical assignment and set partitioning formulations, and is motivated by the poor performance of assignment formulations on difficult instances. Furthermore, we describe a column generation approach to solve the LP-relaxation of all formulations, and show the benefit of a suitably picked formulation using practical instances from NS.

The remainder of this paper is organized as follows. In Section 2, we discuss the general modeling framework for the CCRP. The family of formulations is presented in Section 3. In Section 4, we derive analytical results regarding the tightness of the different formulations. Section 5 describes our computational results for practical instances of NS. The paper is concluded in Section 6.

2 Modeling the Cyclic Crew Rostering Problem

In the CCRP, cyclic rosters have to be constructed for groups of employees. Each cyclic roster consists of rows (i.e., generic work weeks), columns (i.e., weekdays), and cells (i.e., the intersection of a row and a column). An example of a roster is depicted in Figure 1. The roster in Figure 1 is operated by four employees. The first employee performs the first row in Week 1, the second row in Week 2, and so forth. Similarly, the second employee starts in row 2, continues in rows 3 and 4, and then performs row 1 in Week 4. Every four weeks, this process is repeated. As a consequence, all employees in this roster have performed exactly the same work after four weeks.

In the roster, two components are specified: the type specification of each cell (e.g., an early (E), late (L) or night (N) duty, or a day-off (R)), known as the basic schedule, and an allocation of the duties to the cells. We assume the set of duties and the basic schedules to be given (see, e.g., [9] for a discussion on the construction of basic schedules). The output of the CCRP is then a set of rosters in which all duties are assigned.

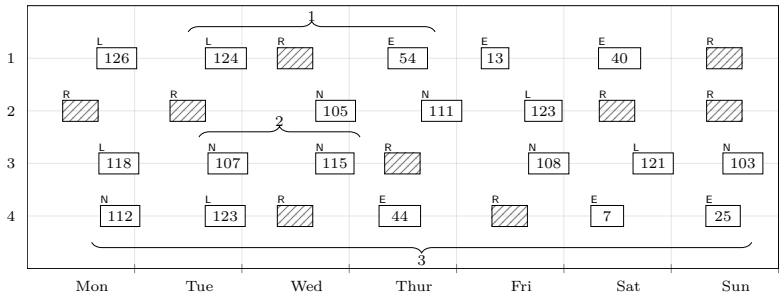


Figure 1 Example of a cyclic roster for a group of four employees. Three roster constraints are indicated. The first roster constraint requires that a scheduled rest period is sufficiently long and the second constraint specifies the minimum time between consecutive duties. The third constraint enforces a maximum workload over a working week.

Two important aspects have to be taken into account when constructing the rosters. Firstly, the roster should be feasible with respect to the labor regulations. For example, there should be sufficient rest time between consecutive duties, and the total amount of work in a row (i.e., in a week of work) cannot be too large. Secondly, the roster should be perceived attractive by the employees. Short, although legal, rest times, for example, make the roster unattractive, as employees prefer a proper rest period between two duties. The feasibility and perceived attractiveness of a roster are expressed using *roster constraints*, which are (linear) constraints depending on the assigned duties: Feasibility (e.g., minimum rest times, maximum workload) is modeled using hard constraints, whereas attractiveness (e.g., undesirable rest times, variation of work) is modeled using soft constraints, thereby penalizing unattractive assignments of duties. In Figure 1, a few roster constraints are highlighted. The first roster constraint, for example, requires that a scheduled rest period (here scheduled on Wednesday), is sufficiently long. In other words, the difference between the end of duty 124 on Tuesday and the start of duty 54 on Thursday should be sufficiently large. The second constraint specifies the minimum time between consecutive duties, assuring that the crew members can have a sufficient rest. Finally, the third constraint considers the work scheduled in an entire row, and could, for example, enforce a maximum workload over a working week.

To obtain a strong formulation for the CCRP, it is important to analyze the types of roster constraints that are present. That is, many roster constraints have a similar structure which should be taken into account when modeling the problem. In Figure 1, for example, the first two roster constraints can be classified as *linking constraints*, i.e., those linking exactly two cells in the basic schedule (note that the rest days are assumed fixed), whereas the third constraint can be classified as a *row-based constraint*. Given such a classification, an efficient modeling of the constraints can be determined, and a strong formulation can be obtained.

In the remainder of this section we discuss the modeling of roster constraints in detail: In Section 2.1, we explain how we model linking constraints. In Section 2.2, we propose a general framework that allows to model many practical roster constraints.

2.1 Modeling Linking Constraints

Linking constraints often occur in crew rostering problems, hence a strong formulation for such constraints can lead to major efficiency gains. We now describe a modeling approach that applies both to hard and soft constraints. In both cases, we assume that the linking constraints are binary, i.e., the constraint is either satisfied or not. For soft constraints, constraint violations are allowed, but in that case a penalty is incurred that is independent of the size of the violation.

Consider a linking constraint between two cells t_1 and t_2. Let D and F denote the respective sets of feasible duties for these cells. Furthermore, let $E \subseteq D \times F$ denote the *violation set* for the linking constraint, i.e., all pairs $(d, f) \in D \times F$ such that assigning d to t_1 and f to t_2 violates the constraint. The linking constraint can be naturally modeled as a bipartite graph. For example, seven duties are depicted as nodes in Figure 2a. For each duty in D (F), the end time (start time) is depicted in the figure as well. Suppose we require a 12 hour rest between two consecutive duties. The corresponding violation graph is a bipartite graph, in which the vertex sets represent the sets of feasible duties D and F, respectively, and an edge $(d, f) \in D \times F$ is present if duties d and f violate the rest time constraint.

To model soft linking constraints, we introduce a decision variable $\delta \in \mathbb{B}$ that indicates whether the linking constraint is violated or not. Furthermore, we define the decision variables $\pi_{t_1 d}$, for $d \in D$, and $\pi_{t_2 f}$, for $f \in F$, indicating whether duty d, respectively f, is assigned to cell t_1, respectively t_2. The linking constraint is readily expressed by the following system of equations.

$$\sum_{d \in D} \pi_{t_1 d} = 1 \tag{1}$$

$$\sum_{f \in F} \pi_{t_2 f} = 1 \tag{2}$$

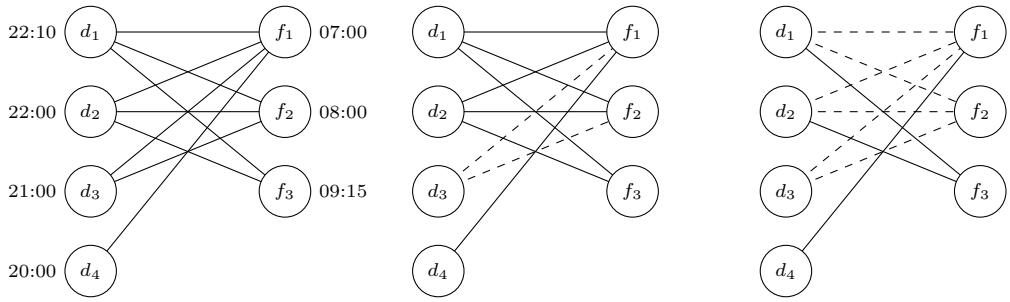
$$\pi_{t_1 d} + \sum_{f \in F:(d,f) \in E} \pi_{t_2 f} \leq 1 + \delta \qquad \forall d \in D \tag{3}$$

$$\delta, \pi_{t_1 d}, \pi_{t_2 f} \in \mathbb{B} \qquad \forall d \in D, f \in F. \tag{4}$$

Constraints (1) and (2) state that exactly one duty should be assigned to both cells and Constraints (3) assure that the constraint violation is modeled correctly. Finally, Constraints (4) give the domains of the decision variables. Note that hard linking constraints can be modeled similarly by forcing $\delta = 0$, or by discarding the decision variable δ altogether.

We will refer to (3) as *flow-based* constraints, as each single constraint sums over the out-going arcs of a single $d \in D$. (Figure 2b visualizes such a constraint.) This type of aggregation has been previously used in [9]. The correctness of (3) is readily seen, as each arc (i.e., violation) appears in exactly one constraint. That is, each combination $d \in D$ and $f \in F$ such that $(d, f) \in E$ appears in exactly one constraint.

Another way of incorporating (3), is based on bicliques in the graph-representation. This type of modeling has been considered in [7] and [11]. To formulate the clique-based constraints, we introduce the following additional notation. For a given $d \in D$, let $D_d \subseteq D$ denote all $d' \in D$ for which $(d, f) \in E$ implies that $(d', f) \in E$ for all $f \in F$. By construction, it always holds that $d \in D_d$. In the case of Figure 2, we have, for example, $D_{d_3} = \{d_1, d_2, d_3\}$, since d_1 and d_2 are also connected with f_1 and f_2, and thus, with all neighbors of d_3. In the case of rest time constraints, D_{d_3} boils down to exactly those duties in D that end at the



(a) Violation graph. **(b)** Flow-based constraint. **(c)** Clique-based constraint.

Figure 2 Example strengthened linking constraints. The dashed edges indicate the variables included in the constraint (either flow- or clique-based) for duty d_3.

same time or later than d_3. The clique-based constraints now read as follows.

$$\sum_{d' \in D_d} \pi_{t_1 d'} + \sum_{f \in F:(d,f) \in E} \pi_{t_2 f} \leq 1 + \delta \qquad \forall d \in D. \tag{5}$$

The clique-based constraints are illustrated in Figure 2c. Note that replacing (3) by (5) is allowed since, by definition of D_d, it holds that $(d', f) \in E$ for all $d' \in D_d$ and $f \in F$ such that $(d, f) \in E$. Furthermore, every violation appears in at least one constraint, since $d \in D_d$. For the rest time constraints that we consider, the number of clique-based constraints is bounded by the number of duties that can be assigned to cell t_1. [5] proves that clique-based constraints lead to the strongest formulation possible for linking constraints.

2.2 General Modeling Framework

We now discuss a general modeling framework for roster constraints. Let D denote the set of duties, T the set of cells, and let D_t denote the duties that can be assigned to cell $t \in T$. Let Q denote the set of roster constraints. Each roster constraint q is modeled using a set of linear constraints $p \in P_q$. Each linear constraint $p \in P_q$ is specified by a coefficient for each assignment of a duty to a cell in the basic schedule, and a scalar called the *threshold value*. The coefficient for the assignment (t, d) for linear constraint p is denoted by f^p_{td} and the threshold value for p is denoted by b_p.

Let δ_q denote the violation of roster constraint q, and let c_q be the corresponding penalty variable. The roster constraints enforce that *if* the sum of coefficients of assigned duties exceeds the threshold value for one of the linear constraints, *then* the difference between the sum and the threshold value is penalized *and* lies within the violation interval, given by $\Delta_q = [0, u_q]$. In other words, the roster constraint is modeled by enforcing each linear constraint $p \in P_q$:

$$\sum_{t \in T} \sum_{d \in D_t} f^p_{td} \pi_{td} \leq b_p + \delta_q, \tag{6}$$

and assuring $\delta_q \in \Delta_q$. Note that (6) assures that δ_q is equal to the maximum violation, calculated over all $p \in P_q$. It is readily seen that both the flow- and clique-based linking constraints fit this framework, and that also many other constraints can be modeled in this fashion.

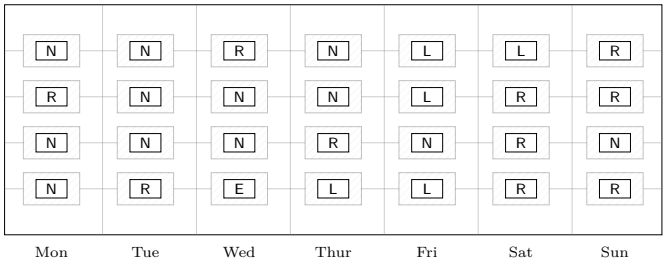
3 Family of Mathematical Formulations

In this section we propose a family of mathematical formulations for the CCRP. In Section 3.1, we define the concept of clusters and roster sequences, and we conclude with a family of mathematical formulations for the CCRP in Section 3.2.

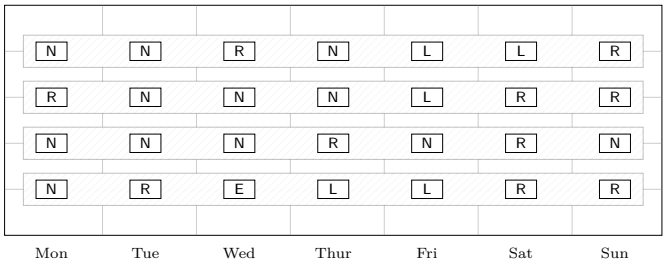
3.1 Clusters and Roster Sequences

The family of formulations is based on different partitions of the basic schedule. That is, we develop a mathematical formulation under the assumption that each basic schedule is partitioned into disjoint subsets, which we call *clusters*. This partition will be referred to as a *clustering* for the respective basic schedule, and should be picked a priori solving the model.

The formulation will have a different structure for each possible clustering, giving rise to the family of formulations. Figure 3 gives an example of two possible clusterings for a basic schedule of four rows. In the cell-based clustering each cluster contains exactly one of the cells in the basic schedule. The row-based clustering, on the other hand, assigns all cells in the same row (i.e., Monday to Sunday) to the same cluster. Note that many more clusterings are possible. One could, for example, also consider a "weekend" clustering, in which each cluster relates to either Friday to Monday (the "weekend" days), or Tuesday to Thursday (the "week" days). Such a clustering can be a good choice when, e.g., the rest time over the weekend is of utmost importance. Generally, cells in a cluster do not need to be consecutive.



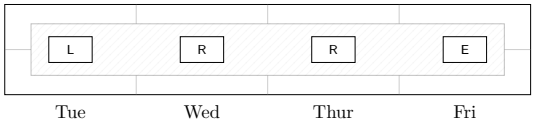
(a) Cell-based clustering.



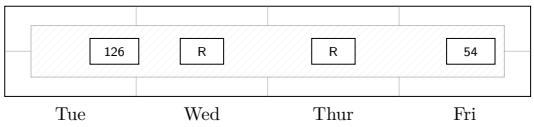
(b) Row-based clustering.

Figure 3 Example of different clusterings. Each highlighted area represents a cluster.

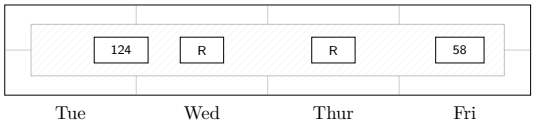
Each cluster is assigned a number of duties simultaneously. Each possible assignment of duties to a cluster is called a *roster sequence*. Formally, a roster sequence specifies a duty or rest day for each cell in the cluster, such that the assignment is compatible with the basic schedule, and such that no duty is assigned twice (within the same cluster).



(a) Cluster.



(b) Roster Sequence 1.



(c) Roster Sequence 2.

Figure 4 Cluster from Tuesday to Friday, together with two possible roster sequences.

To illustrate the use of roster sequences, consider the cluster depicted in Figure 4, together with two possible roster sequences. Note that the roster sequences contain different duties (indicated by the numbers). In this case the second roster sequence has a shorter rest period than the first roster sequence (as duty 124 ends later than duty 126, and duty 58 starts earlier than duty 54), which might be considered undesirable.

The goal of a clustering is to model constraints implicitly using the roster sequences. That is, ideally each constraint considers the cells in *solely* one of the clusters, and can therefore be taken care of when generating the roster sequences. As an example, consider a constraint in which an employee can have only a maximum amount of work per row. In this case, the row-based clustering of Figure 3 allows to model these constraints implicitly using the roster sequences (i.e., a roster sequence is feasible only if it does not exceed the maximum working time). On the other hand, for the cell-based clustering these constraints have to be modeled explicitly in the mathematical formulation.

3.2 Mathematical Formulation

We are now ready to formalize the family of formulations. The set K denotes the set of all clusters (note that these are determined a priori formulating the mathematical model). We define the set S_k as the set of all roster sequences for cluster $k \in K$. Each roster sequence can be seen as a sequence of assignments (t, d). The parameter h^k_{ds} indicates whether roster sequence $s \in S_k$ contains duty d (i.e., duty d appears in one of the assignments describing the roster sequence s). Finally, we define c^k_s as the penalty associated with roster sequence $s \in S_k$ for cluster $k \in K$.

Let $Q_k \subseteq Q$ denote the set of roster constraints fully contained in cluster $k \in K$, and define $Q_K = \bigcup_{k \in K} Q_k$. The constraints in Q_K are exactly those that are modeled implicitly using the roster sequences. The penalty c^k_s associated with roster sequence $s \in S_k$ is the sum of all violations in the roster sequence s, restricted to the roster constraints Q_k. Note that the roster constraints in $Q \setminus Q_K$ need to be modeled explicitly.

To model the CCRP, given a clustering K, we introduce the following decision variables.

- x^k_s, for all $k \in K$ and $s \in S_k$. The binary variable x^k_s indicates whether roster sequence $s \in S_k$ is assigned to cluster $k \in K$.

- δ_q, for each $q \in Q \setminus Q_K$. The variable $\delta_q \in \Delta_q$ expresses the violation of the roster constraint $q \in Q \setminus Q_K$.

The formulation now reads as follows.

$$\min \quad \sum_{k\in K} \sum_{s\in S_k} c_s^k x_s^k + \sum_{q\in Q\setminus Q_K} c_q \delta_q \tag{7}$$

$$\text{s.t.} \quad \sum_{s\in S_k} x_s^k = 1 \qquad \forall k \in K \tag{8}$$

$$\sum_{k\in K} \sum_{s\in S_k} h_{ds}^k x_s^k = 1 \qquad \forall d \in D \tag{9}$$

$$\sum_{k\in K} \sum_{s\in S_k} \sum_{(t,d)\in s} f_{td}^p x_s^k \leq b_p + \delta_q \qquad \forall q \in Q \setminus Q_K, p \in P_q \tag{10}$$

$$x_s^k \in \mathbb{B} \qquad \forall k \in K, s \in S_k \tag{11}$$

$$\delta_q \in \Delta_q \qquad \forall q \in Q \setminus Q_K. \tag{12}$$

The Objective (7) minimizes the penalties for violating soft roster constraints. The first term corresponds to the soft roster constraints that are modeled implicitly. In particular, the roster sequence costs can be expressed as

$$c_s^k = \sum_{q\in Q_k} c_q \bar{\delta}_q,$$

where the constraint violation $\bar{\delta}_q$ for $q \in Q_k$ can be computed directly from the roster sequence $s \in S_k$, by definition of Q_k. The second term equals the penalties for all soft roster constraints that are modeled explicitly.

Constraints (8) and (9) assure that the duties are assigned correctly to the basic schedules. That is, each cluster is assigned exactly one roster sequence, and each duty is assigned exactly once to a cell in the basic schedule. Constraints (10) represent the roster constraints that are modeled explicitly. Finally, Constraints (11) and (12) specify the domains of the decision variables. The family of formulations for the CCRP is now obtained by taking (7)–(12) for all possible clusterings K. The family includes the generalized assignment model and set partitioning formulation from literature. First, the cell-based clustering, depicted in Figure 3a, gives rise to the generalized assignment model that has been applied in [9]. In contrast, by viewing the complete time horizon as one cluster, a set partitioning formulation is obtained.

Our aim is to analyze the family of formulations by considering the relative strength of its members. For coarser clusterings, the number of roster sequences can be huge. Therefore, we now describe a column generation approach to solve the LP-relaxation of the CCRP formulation (7)–(12). The master problem is obtained from (7)–(12) by relaxing the integrality constraints on the x_s^k variables. The reduced cost γ_s^k of a roster sequence $s \in S_k$, for a given cluster $k \in K$ can be expressed as follows. Let μ_k denote the dual variables corresponding to (8), ϕ_d those corresponding to (9), and θ_{qp} those corresponding to (10). The reduced cost γ_s^k can now be expressed as

$$\gamma_s^k = c_s^k - \mu_k - \sum_{d\in D} h_{ds}^k \phi_d - \sum_{q\in Q\setminus Q_K} \sum_{p\in P_q} \sum_{(t,d)\in s} f_{td}^p \theta_{qp}.$$

For each $k \in K$, the pricing problem can be modeled as a resource constrained shortest path problem (RCSPP) with surplus variables on a directed layered graph $G_k = (V_k, A_k)$ (see [8, 10]). In this graph, each vertex corresponds to an assignment (t, d) of a duty to a cell in

k and each arc corresponds to a feasible follow-up of two assignments. Note that the implicit roster constraint penalty for $q \in Q$ has to be taken into account by taking the maximum over all $p \in P_q$.

4 Theoretical Comparison Clusterings

Intuitively, the implicit modeling of the roster constraint violations leads to a tighter linear relaxation. In this section, we prove this rigorously. From hereon, we consider two clusterings K and L. We show that the strength of the linear relaxation depends on Q_K and Q_L, and not necessarily on K and L, i.e., shifting from K to L will not change the root bound if $Q_K = Q_L$. This provides a systematic way to identify candidate clusterings, i.e., clusterings that potentially improve the objective value of the LP-relaxation.

From hereon, we assume that $Q_K \supseteq Q_L$. An example of two clusterings for which this holds is given in Figure 3, where K and L are the row-based and cell-based clustering, respectively. Let S_k, for all $k \in K$, and G_ℓ, for all $\ell \in L$, denote the respective sets of roster sequences for both clusters. For notational convenience, define Ω as the set of all feasible assignments (t, d), with $t \in T$ and $d \in D_t$, of duties to the cells in the basic schedule. Furthermore, we define the operator $[\cdot]^+$ as $[a]^+ = \max\{0, a\}$. Throughout this section, a *solution* refers to a solution to the linear relaxation.

We first state the following lemma. Intuitively, this lemma states that, given a solution for K, we can construct a solution for L such that each duty is assigned to the same cell in both solutions. The proof of this lemma can be found in Appendix A.

▶ **Lemma 1.** *Let $\bar{x}$ be a solution for clustering K. If $Q_K \supseteq Q_L$, then there exists a feasible solution $\bar{z}$ for clustering L, such that*

$$\sum_{k \in K} \sum_{\substack{s \in S_k: \\ s \ni (t,d)}} \bar{x}_s^k = \sum_{\ell \in L} \sum_{\substack{g \in G_\ell: \\ g \ni (t,d)}} \bar{z}_g^\ell \tag{13}$$

for each $(t, d) \in \Omega$.

It is important to note that Lemma 1 does not hold in the opposite direction. That is, given a solution $\bar{z}$ it is not always possible to construct a solution $\bar{x}$ satisfying (13). Furthermore, note that we only require that $Q_K \supseteq Q_L$. The clustering K being coarser than L is a sufficient, but not a necessary condition for this to hold.

We are now able to prove the following theorem. For the proof of this theorem, we again refer to Appendix A.

▶ **Theorem 2.** *Let K and L be two clusterings such that $Q_K \supseteq Q_L$. Furthermore, let v_K denote the optimal value of the LP-relaxation using clustering K, and define v_L similarly. Let $\bar{x}$ be an optimal solution corresponding to v_K. It holds that*

$$v_K \geq v_L + \sum_{q \in Q_K \setminus Q_L} c_q \phi_q(\bar{x}),$$

where the non-negative coefficients $\phi_q(\bar{x})$ are given by

$$\phi_q(\bar{x}) = \sum_{k \in K} \sum_{s \in S_k} \bar{x}_s^k \cdot \max_{p \in P_q} \left[\sum_{(t,d) \in s} f_{td}^p - b_p \right]^+ - \max_{p \in P_q} \left[\sum_{k \in K} \sum_{s \in S_k} \bar{x}_s^k \left(\sum_{(t,d) \in s} f_{td}^p \right) - b_p \right]^+ .$$

The value $\phi_q(\bar{x})$ represents the error incurred from modeling roster constraint q explicitly (note that $\phi_q(\bar{x})$ is zero if $\bar{x}$ is integer), opposed to modeling it implicitly (i.e., correctly). Theorem 2 can be used as a guideline to pick the "ideal" set of clusters. In particular, the proof of Theorem 2 leads to two key insights. The first one is formalized in the following corollary.

▶ **Corollary 3.** *Let K and L be two clusterings such that $Q_K = Q_L$. Denoting v_K for the optimal value of the LP-relaxation using clustering K, and v_L similarly, it holds that $v_K = v_L$.*

The corollary shows that switching from clustering L to K with K coarser than L, i.e, every $\ell \in L$ is a subset of some $k \in K$, but $Q_K = Q_L$ is never beneficial, i.e., will not increase the LP-bound. This implies that the roster constraints should be explicitly considered when enlarging the cluster size.

Secondly, the theorem shows that switching from L to K is likely to be beneficial whenever $Q_K \setminus Q_L$ contains "weak" roster constraints, where the weakness is represented by the value of $c_q\phi_q(\bar{x})$. Note that, although this value is not known a priori, it is often possible to estimate these values based on, e.g., experience or expert knowledge.

5 Computational Experiments

In this section we discuss the computational results. We first discuss the experimental set-up in Section 5.1. That is, we discuss the roster constraints that are taken into account, and the different instances considered. We then present the computational results in Section 5.2.

5.1 Experimental Set-Up

We apply our solution approach to different instances based on data from NS. For each instance, the basic schedule specifies the days off. Furthermore, for each duty that is to be scheduled a type is given. The considered types are Early, Late, and Night. The type of each duty is based on the start time of the duty. The following roster constraints are taken into account.

- *Rest Time.* After completing a duty it is required that an employee has a certain minimum time to rest. After a night duty this rest time should be at least 14 hours, otherwise it should be at least 12 hours. Furthermore, we penalize rest times shorter than 16 hours with a penalty of 30.
- *Rest Day.* When rest days are scheduled in the roster, the length of the rest period has to be sufficient. This implies that there is a minimal time enforced between duties scheduled before and after the rest days. The enforced rest time is 6 hours plus 24 hours for each rest day.
- *Red Weekend.* At least once every three rows of the roster there should be a weekend which has a consecutive period of 60 hours off. These so-called red weekends can be determined given the basic schedule. The 60 hour rest period can then be enforced using the roster constraints.
- *Workload.* The total workload in a row is not allowed to exceed 45 hours. Here, the workload of a duty is the difference between the start and end time (i.e., including the meal break).
- *Variation.* The variation constraints assure that the different attributes of work (e.g., duty length, percentage double decker work) are divided equally over the rows. These constraints penalize a positive deviation from the average (measured over all duties) for each row in the roster. In total we consider 10 different variation constraints.

We consider a total of 10 different instances: four "small" instances of 12 employees and roughly 50 duties, four "medium" instances of 24 employees and roughly 100 duties, and two "large" instances of about 50 employees and 200 duties. Each of the instances is obtained by combining multiple roster groups as operated at NS. The properties of the instances are summarized in Table 1. At NS, the rostering problem for a medium-sized crew base has similar characteristics as instance 9.

Table 1 Characteristics of the instances. For each instance the number of groups and number of employees (i.e., the number of rows) is specified, along with the number of Early, Late, and Night duties, and the total number of duties.

	Groups	Employees	Early	Late	Night	Total
1	1	12	23	11	15	49
2	1	12	21	12	16	49
3	1	12	49	0	1	50
4	1	12	49	0	1	50
5	2	24	21	36	35	92
6	2	24	23	35	37	95
7	2	26	101	0	2	103
8	2	24	97	0	2	99
9	4	54	118	47	52	217
10	4	50	198	0	4	202

The instances can be categorized into one of two categories. The instances 1, 2, 5, 6, and 9 represent instances in which all three duty types have to be scheduled. For the other instances the duties consist almost exclusively of early duties. The former category of instances provide more structure compared to the latter ones, since (i) less roster sequences are possible (as the duties are divided over different types), and (ii) the rest time and rest day constraints are expected to be more important for these instances (i.e., if all duties start early, the chance of having a rest time violation is small). The second category is therefore expected to be more difficult to solve if the formulation is not chosen carefully.

5.2 Computational Results

In this section the computational results are discussed in detail. In particular, we compare the performance of different clusterings and evaluate the modeling of linking constraints. All experiments are done on a computer with a 1.6 GHz Intel Core i5 processor. We use the LP-solver embedded in CPLEX 12.7.1 to solve the restricted master problems.

To illustrate the effect of different clusterings (for the given constraints) and the modeling of linking constraints, we solve the LP-relaxation for four clusterings and both the flow- and clique-based linking constraints. We consider clusterings where each cluster contains a single cell, three cells, six cells, and seven cells (i.e., a cluster per row). We denote these clusterings by C_1, C_3, C_6, and C_7, respectively. Each clustering leads to a different formulation. In particular, the clustering C_1 results in the assignment formulation proposed in [9], and the clustering C_7 leads to the row-based formulation used in [4].

Table 2 shows for each clustering and each instance, the objective value of the LP-relaxation for the flow- and clique-based constraints, together with the percentage of constraints that can be modeled implicitly. (The non-zero percentage for C_1 and instance 6 is due to one row in which only one duty has to be assigned.) The results in Table 2 are in line with Theorem 2. That is, there is a clear relation between the percentage of implicit constraints

Table 2 Comparison of different clusterings and the modeling of linking constraints. For each clustering and each instance, the root bound for the flow- and clique-based constraints are shown, together with the percentage of constraints that can be modeled implicitly.

		1	2	3	4	5	6	7	8	9	10
	Flow	558.0	654.4	192.4	274.0	671.3	618.3	181.0	250.4	916.9	221.4
C_1	Clique	570.1	681.8	192.8	286.5	833.1	843.1	302.8	345.9	1213.2	330.5
	Impl. (%)	0.0	0.0	0.0	0.0	0.0	1.7	0.0	0.0	0.0	0.0
	Flow	569.3	660.9	192.5	289.5	762.6	800.5	196.7	297.9	1103.7	251.9
C_3	Clique	571.4	687.4	192.8	299.6	850.0	889.8	309.6	370.6	1245.8	351.4
	Impl. (%)	29.1	20.2	31.6	38.2	37.7	39.7	42.4	50.7	48.6	53.6
	Flow	581.3	705.5	206.4	313.5	834.1	825.8	256.9	340.4	1192.6	301.5
C_6	Clique	584.1	705.8	206.4	318.0	875.3	914.8	335.7	396.9	1278.6	390.6
	Impl. (%)	49.3	59.0	49.3	59.4	61.3	56.0	64.5	68.0	67.6	73.4
	Flow	609.8	713.8	280.0	370.2	873.4	943.2	447.8	523.4	1312.7	598.0
C_7	Clique	609.8	713.8	280.0	370.2	942.7	1004.0	447.8	523.4	1435.0	598.0
	Impl. (%)	98.0	94.7	94.5	98.6	92.3	93.2	91.4	94.6	93.8	92.0

and the bound obtained from the LP-relaxation. The benefit of a suitable clustering is most apparent for the instances with mostly early duties (i.e., instances 3, 4, 7, 8, and 10). For these instances the main challenge is to capture the cost incurred from the variation constraints, which only clustering C_7 is able to do. Furthermore, we see that the clique-based linking constraints improve the LP-bound substantially for the mixed instances (i.e., those with relatively many rest time violations). If we consider C_7, for example, we see that these constraints substantially improve the root bound for almost all instances with mixed duty types, namely for instances 5, 6, and 9. Only for the smaller mixed instances 1 and 2 no improvement is found. Note that this improvement is expected for the mixed instances, as opposed to the non-mixed instances, where rest time violations hardly occur.

In order to find integer solutions to the CCRP, our column generation algorithm can be embedded in a Branch-and-Price framework. Computational results in [5] show that clustering C_7 also leads to better integer solutions in a short amount of time, compared to the other clusterings.

6 Conclusion

In this paper, we analyzed formulations for the Cyclic Crew Rostering problem (CCRP), in which attractive cyclic rosters have to be constructed for groups of employees. We proposed a family of formulations, motivated by the poor performance of traditional assignment models for difficult instances. Each formulation has a different structure, which implies that a suitable variant can be picked for a given problem instance. We derived analytical results regarding the relative strength of the different formulations, which can be used as a guideline to pick a suitable formulation for a given problem instance.

We also developed a column generation approach to solve the LP-relaxation of each formulation in the family. The pricing of columns in this approach is done by solving a resource constrained shortest path problem (RCSPP) with surplus variables. We applied our method to practical instances from NS. Our experiments showed the importance of picking a suitable formulation for a given problem instance. In particular, we show that a suitable formulation is better able to capture the penalty incurred from the roster constraints. Furthermore, we showed that the clique-based modeling of linking constraints improves the LP-bound substantially.

References

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

2 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.

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

4 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.

5 Thomas Breugem. *Crew Planning at Netherlands Railways: Improving Fairness, Attractiveness, and Efficiency*. PhD thesis, Erasmus University Rotterdam, January 2020.

6 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.

7 A. T. Ernst, H. Jiang, M. Krishnamoorthy, H. Nott, and D. Sier. An integrated optimization model for train crew management. *Annals of Operations Research*, 108(1-4):211–224, 2001.

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

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

10 S. Irnich and G. Desaulniers. Shortest path problems with resource constraints. In *Column Generation*, pages 33–65. Springer, 2005.

11 M. Mesquita, M. Moz, A. Paias, and M. Pato. A decompose-and-fix heuristic based on multi-commodity flow models for driver rostering with days-off pattern. *European Journal of Operational Research*, 245(2):423–437, 2015.

12 M. Sodhi and S. Norris. A flexible, fast, and optimal modeling approach applied to crew rostering at London Underground. *Annals of Operations Research*, 127(1-4):259–281, 2004.

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

A Proofs

▶ **Lemma 1.** *Let $\bar{x}$ be a solution for clustering K. If $Q_K \supseteq Q_L$, then there exists a feasible solution $\bar{z}$ for clustering L, such that*

$$\sum_{k \in K} \sum_{\substack{s \in S_k: \\ s \ni (t,d)}} \bar{x}_s^k = \sum_{\ell \in L} \sum_{\substack{g \in G_\ell: \\ g \ni (t,d)}} \bar{z}_g^\ell \tag{13}$$

for each $(t, d) \in \Omega$.

Proof. We consider an auxiliary clustering O, defined as the coarsest clustering which is finer than both K and L (see Figure 5). Formally, O is uniquely defined by taking all non-empty subsets $k \cap \ell$, with $k \in K$ and $\ell \in L$. Let R denote the set of feasible roster sequences for this clustering, and let R_o denote the feasible roster sequences for $o \in O$.

Since each cluster $o \in O$ is fully contained in some $k \in K$, we can readily obtain a solution $\bar{y}$ for O satisfying

$$\sum_{k \in K} \sum_{\substack{s \in S_k: \\ s \ni (t,d)}} \bar{x}_s^k = \sum_{o \in O} \sum_{\substack{r \in R_o: \\ r \ni (t,d)}} \bar{y}_r^o \tag{14}$$

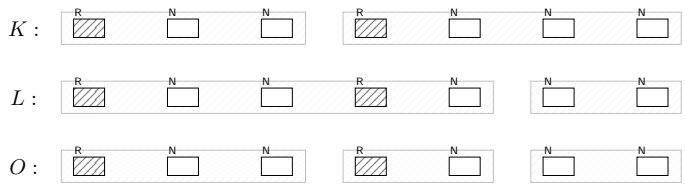


Figure 5 Example of the clustering O, which is the coarsest clustering finer than both K and L.

by splitting up each roster sequence for K into smaller roster sequences for O. Furthermore, since each $o \in O$ is also fully contained in some $\ell \in L$, we can obtain a solution $\bar{z}$ satisfying

$$\sum_{\ell \in L} \sum_{\substack{g \in G_\ell: \\ g \ni (t,d)}} \bar{z}_g^\ell = \sum_{o \in O} \sum_{\substack{r \in R_o: \\ r \ni (t,d)}} \bar{y}_r^o \tag{15}$$

by greedily constructing roster sequences for L given those for O. To be more precise, let $O_\ell \subseteq O$ denote the clusters contained in $\ell \in L$. For each $\ell \in L$, we pick the roster sequence r with smallest non-zero value $\bar{y}_r^o$, say v, over all clusters in O_ℓ. This roster sequence is then combined with a roster sequence for each of the other clusters in O_ℓ, to obtain a roster sequence g for cluster ℓ. We set $\bar{z}_g^\ell = v$, reduce $\bar{y}_r^o$ for all involved roster sequences by v, and repeat the procedure until all roster sequences are assigned.

It follows that we can construct a solution $\bar{z}$ that satisfies (13). It remains to show that a solution constructed in this fashion is feasible with respect to the roster constraints.

We first show that $\bar{z}$ is feasible for the roster constraints in $Q \setminus Q_L$. Consider some $q \in Q \setminus Q_L$ and fixed $p \in P_q$. Recall that u_p is the upper bound of the violation interval Δ_p. Using (13) we have

$$\sum_{\ell \in L} \sum_{g \in G_\ell} \sum_{(t,d) \in g} f_{td}^p \bar{z}_g^\ell = \sum_{(t,d) \in \Omega} \sum_{\ell \in L} \sum_{\substack{g \in G_\ell: \\ g \ni (t,d)}} f_{td}^p \bar{z}_g^\ell \tag{16a}$$

$$= \sum_{(t,d) \in \Omega} \sum_{k \in K} \sum_{\substack{s \in S_k: \\ s \ni (t,d)}} f_{td}^p \bar{x}_s^k \tag{16b}$$

$$= \sum_{k \in K} \sum_{s \in S_k} \sum_{(t,d) \in s} f_{td}^p \bar{x}_s^k. \tag{16c}$$

Hence, for $q \in Q \setminus Q_K$ and $p \in P_q$, the feasibility of $\bar{x}$ implies that

$$\sum_{\ell \in L} \sum_{g \in G_\ell} \sum_{(t,d) \in g} f_{td}^p \bar{z}_g^\ell - b_p = \sum_{k \in K} \sum_{s \in S_k} \sum_{(t,d) \in s} f_{td}^p \bar{x}_s^k - b_p \leq u_p. \tag{17}$$

Next, consider some $q \in Q_K \setminus Q_L$ and $p \in P_q$. Since $q \in Q_K$, there is a cluster $k' \in K$ such that the coefficients f_{td}^p are non-zero only for this cluster. It follows that

$$\sum_{\ell \in L} \sum_{g \in G_\ell} \sum_{(t,d) \in g} f_{td}^p \bar{z}_g^\ell - b_p = \sum_{k \in K} \sum_{s \in S_k} \sum_{(t,d) \in s} f_{td}^p \bar{x}_s^k - b_p \tag{18a}$$

$$= \sum_{s \in S_{k'}} \bar{x}_s^{k'} \sum_{(t,d) \in s} f_{td}^p - b_p \tag{18b}$$

$$= \sum_{s \in S_{k'}} \bar{x}_s^{k'} \left(\sum_{(t,d) \in s} f_{td}^p - b_p \right), \tag{18c}$$

where (18c) follows from (8). Using the feasibility of the roster sequence s, we have

$$\sum_{s \in S_{k'}} \bar{x}_s^{k'} \left(\sum_{(t,d) \in s} f_{td}^p - b_p \right) \leq \sum_{s \in S_{k'}} \bar{x}_s^{k'} u_p \tag{19a}$$

$$= u_p, \tag{19b}$$

where (19b) follows from (8). It follows that $\bar{z}$ is feasible for all $q \in Q_K \setminus Q_L$, and thus for all $q \in Q \setminus Q_L$.

To show that $\bar{z}$ is feasible with respect to the roster constraints in Q_L we make the following crucial observation: Since $Q_K \supseteq Q_L$ it must hold that $Q_O \supseteq Q_L$, and hence $Q_O = Q_L$. Suppose that this would not be true, then there must be a roster constraint $q \in Q_L$ and linear constraint $p \in P_q$ with non-zero coefficient f_{td}^p for multiple clusters in O. By definition of O, however, this would imply that $Q_L \setminus Q_K \neq \emptyset$, as O is the coarsest clustering finer than both K and L. This contradicts the assumption that $Q_K \supseteq Q_L$. Hence, if the constructed solution $\bar{y}$ is feasible with respect to Q_O, then a solution $\bar{z}$ created by combining these roster sequences must be feasible with respect to Q_L. The feasibility of $\bar{y}$ with respect to Q_O, however, follows directly from the feasibility of $\bar{x}$, since $Q_O \subseteq Q_K$. This concludes the proof. ◀

▶ **Theorem 2.** *Let K and L be two clusterings such that $Q_K \supseteq Q_L$. Furthermore, let v_K denote the optimal value of the LP-relaxation using clustering K, and define v_L similarly. Let $\bar{x}$ be an optimal solution corresponding to v_K. It holds that*

$$v_K \geq v_L + \sum_{q \in Q_K \setminus Q_L} c_q \phi_q(\bar{x}),$$

where the non-negative coefficients $\phi_q(\bar{x})$ are given by

$$\phi_q(\bar{x}) = \sum_{k \in K} \sum_{s \in S_k} \bar{x}_s^k \cdot \max_{p \in P_q} \left[\sum_{(t,d) \in s} f_{td}^p - b_p \right]^+ - \max_{p \in P_q} \left[\sum_{k \in K} \sum_{s \in S_k} \bar{x}_s^k \left(\sum_{(t,d) \in s} f_{td}^p \right) - b_p \right]^+.$$

Proof. Let $\bar{z}$ be a feasible solution for clustering L satisfying (13), obtained using the construction heuristic described in the proof of Lemma 1. Note that $\bar{z}$ is feasible for L and hence the objective value of $\bar{z}$ is an upper bound for v_L. Furthermore, note that, by the construction of $\bar{z}$, the cost incurred for the roster constraints Q_L is identical for $\bar{x}$ and $\bar{z}$. As a consequence, the difference in objective value between $\bar{x}$ and $\bar{z}$ is exactly the penalty incurred by the roster constraints in $Q_K \setminus Q_L$. Hence, the difference in the penalty incurred by these constraints is a lower bound on $v_K - v_L$.

First, consider the solution $\bar{x}$. Recall that the constraint violations for each pattern $q \in Q_K \setminus Q_L$ are modeled implicitly in the roster sequence cost for clustering K. The penalty incurred from roster constraint $q \in Q_K \setminus Q_L$ is therefore given by

$$c_q \sum_{k \in K} \sum_{s \in S_k} \bar{x}_s^k \cdot \max_{p \in P_q} \left[\sum_{(t,d) \in s} f_{td}^p - b_p \right]^+.$$

Next, consider the solution $\bar{z}$. Note that for L the constraint violations for all $q \in Q_K \setminus Q_L$ are modeled explicitly using (10). Hence, the penalty incurred from roster constraint $q \in Q_K \setminus Q_L$

is given by

$$c_q \max_{p \in P_q} \left[\sum_{\ell \in L} \sum_{g \in G_\ell} \bar{z}_g^\ell \left(\sum_{(t,d) \in g} f_{td}^p \right) - b_p \right]^+ .$$

Using that

$$\sum_{\ell \in L} \sum_{g \in G_\ell} \sum_{(t,d) \in g} f_{td}^p \bar{z}_g^\ell = \sum_{k \in K} \sum_{s \in S_k} \sum_{(t,d) \in s} f_{td}^p \bar{x}_s^k,$$

(see (16)), it follows that the difference in incurred penalty is given by

$$c_q \sum_{k \in K} \sum_{s \in S_k} \bar{x}_s^k \cdot \max_{p \in P_q} \left[\sum_{(t,d) \in s} f_{td}^p - b_p \right]^+ - c_q \max_{p \in P_q} \left[\sum_{k \in K} \sum_{s \in S_k} \bar{x}_s^k \left(\sum_{(t,d) \in s} f_{td}^p \right) - b_p \right]^+ .$$

The result now follows from summing over all $q \in Q_K \setminus Q_L$. ◀