University of Paderborn
Faculty of Business and Economics
Depatment of Information Systems
Decision Support& Operations Research Lab

Working Papers

WP1308

A Column Generation Approach for the Cyclic and Non-cyclic Bus Driver Rostering Problems

Lin Xie

Paderborn, Germany
November, 2013

1. Introduction.

This paper presents a method for optimally solving the problem of constructing monthly, individual work schedules for each driver or for each group of drivers in public transportation companies. The generated work schedules are called *rosters*. In this problem, known as *crew rostering*, monthly crew schedules must be constructed by assigning duties, days off, and other activities to drivers, while taking the complex law and labor union rules into account, such as the minimum rest period between two duties or the maximum number of consecutive working days. A *duty* is a sequence of tasks within one day that is performed by a driver who leaves and returns to the same depot in accordance with the work regulations. Work regulations include maximum duration of a duty, minimal break times during a duty. Such duties are generated in the *crew scheduling problem*, which is the previous step of the crew rostering problem. Both problems are important and difficult problems for bus companies, since the expenses for the drivers and other personal represent a significant portion of bus operators' budgets (more than 50%). Moreover, complex regulations should be considered during the optimization. Therefore, it is common in the public bus transit literature to solve both problems one after the other, i.e., the crew rostering problem uses duties generated in the crew scheduling problem as input.

Compared to the crew scheduling problem, the crew rostering problem has received much less attention in academic literature, since most of the cost benefit can be achieved by minimizing the needed duties in the crew scheduling problem. However, the minimization of costs is still important in the crew rostering problem, and the preferences of drivers are considered during the optimization as well. The rosters which are generated by considering desires of drivers bring higher acceptance than rosters that ignore individual wishes (see Hanne et al. (2009)), which might cause less exchanges and less absence in operational days. Therefore, less recovery activities are expected, which implies lower operational costs, and better services are expected.

Because the crew rostering problem is complex, most publications concentrate on solving its sub-problems in public bus transit. Moreover, most methods proposed in the literature for solving the whole rostering problem rely on heuristics or metaheuristics. Fewer publications discuss the application of column generation for solving the crew rostering problem in public bus transit. Based on the multi-commodity network as well as mathematical models presented in Xie and Suhl (2013), this paper proposes a new solution approach, column generation, for solving the cyclic and non-cyclic crew rostering problems (see Section 2.2) in public bus transit. The tests were conducted on problems from German bus companies. The results experimentally prove to outperform the exact solution approach in Xie and Suhl (2013), as well as different meta heuristics shown in Xie et al. (2013) in terms of solution quality.

The paper is organized as follows. In Section 2, the crew rostering problem is introduced, followed by a brief review of the literature in Section 3, as well as a short description of our network design (Section 4). Section 5 and 6 explain the column generation approach for solving both cyclic and non-cyclic crew rostering problems. The last section presents our results.

2. Crew Rostering Problem.

In this section, we first give details about the input information for solving the crew rostering problem. Then a definition of cyclic and non-cyclic crew rostering problems is described.

2.1. Input information

Different kinds of input information for solving the crew rostering problem, including the information of duties, other activities, drivers, as well as rules and regulation are shown below.

Duty information

Each generated *duty* in the crew scheduling problem has a beginning time, ending time, paid work duration (not necessarily equal to the ending time minus beginning time), the calendar day/weekday it belongs to, the *shift type* (e.g., an early shift, a midday shift, or a late shift), and the vehicle and depot types. The duties need to be assigned to a group of possible drivers who work at the depot of those duties and can drive the bus with the required vehicle type.

Other activities

Not only the duties/shifts must be assigned to drivers, but also some other activities, such as standbys, days off, leaves, and trainings. *Days off* consists of a couple of rest days between working days. *Standby* activities are planned to cover the absences of drivers, while *leaves* are vacations. The trainings and leaves are preassigned and fixed for each driver and cannot be changed in the optimization. The distribution of them is given by bus companies and drivers.

Driver information

The data of drivers not only include the depot and vehicle types, but also the number of the days off, the target working hours for the current planning period, the target number of standby activities as well as the required trainings and leaves. All of those depend on the work contracts and the current work-accounts of the drivers. The work-account of each driver includes the current overtime and the number of days off from the previous periods. It describes the credit of a driver, e.g., a driver with more overtime in the previous periods can get more jobs with shorter working hours in order to reduce overtimes gradually. Additionally, the drivers can express their preferences, including the daily desired activities and the possible combination of activities. The daily desire of a driver means that the driver wishes to get one activity on one day, while the possible combination of activities can be for instance similar shifts within a working week, and/or not to get an early shift after

days off. Hanne et al. (2009) explain the importance of considering the preferences of employees in scheduling their working time, in particular, the advantages of assigning flexible working times to employees. They mention that the employees can better manage their work-life balances, which brings increased morale and reduced absenteeism. That means less exchanges and less absence in operational days, which implies lower operational costs, since less recovery activities are expected. Moreover, they assume that happy staff means happy customers. Better services are expected.

Rules and regulations

The rules and regulations can be labor rules, some of which are imposed by the bus companies, and others are due to the agreement between the bus company and employee unions. Every bus company can also define its own internal constraints that are stricter than those required by law. We consider three types of rules: horizontal rules, vertical rules, and quality rules. Horizontal rules are the rules, which depend only on one roster, while vertical rules combine the information among all rosters. Quality rules are the horizontal rules, without these rules the generated rosters is legal, but they affect the quality of the rosters. Compared to the rules in airline sector in Kohl and Karisch (2004), we have more complex horizontal rules (due to complex rules to generate a feasible roster), but less complex vertical rules (since drivers work mostly alone).

Horizontal rules consider compatibility, working/days off block. The *incompatible* connections of activities are gathered in a list of forbidden sequences according to, for example, the working regulations. An early shift is forbidden to follow a late shift because of the short of rest times between them. The forbidden sequences are given by bus companies, and it is possible that an undesired combination of activities of drivers are included to be forbidden as well. The length of forbidden sequences is not limited to two; for instance, a late shift one day before a standby and an early shift one day later may be forbidden if they appear simultaneously on three consecutive days. Additionally, two other situations are also considered: double-off is defined as at least two consecutive days off, while a single-off is a single day off between two work-related activities. Many of the restrictions in this class can be implicitly considered during the network generation (see Xie and Suhl (2013)). The maximum consecutive number of working days and days off (*working/days off block*) are restricted based on working regulations. For example, the length of the consecutive working days is limited to five, while the length of the consecutive days off is restricted to three.

Vertical rules consider restricted resources among all rosters. Not only the amount of duty/shift of each day is limited to be assigned, but the available numbers of days off and standbys are also limited. The limit depends on the number of drivers and duties available on that day. The upper bound can be easily computed before optimization.

The horizontal and vertical rules described above are formulated as hard constraints in Xie and Suhl (2013), while the following quality rules are formulated as soft constraints. They can be suspended by applying specific penalty costs in the objective to restrict the amount of violations.

Quality rules include preferences of bus company and drivers, real operational costs, fairness among all drivers, as well as robustness of the rosters.

Preferences: in this category, the preferences of the bus company as well as the drivers are considered. As mentioned before, the daily desired activities and the possible combination of activities are given by drivers. Some alternatives can be provided by the bus company if the primary desires of the drivers cannot be satisfied. The number of alternatives, as well as the penalty costs of them can vary from one bus company to another (see Example 1).

Example 1. Bus company A and B have four types of shifts (early1, early2, midday, late shift). Early2 (ES2) begins a couple hours later than early1 (ES1) but earlier than midday. In both bus companies, 50 % of drivers want to get eary shifts (regardless of early1 or early2) and 40 % of them want to get midday shifts (MS). Late shifts are unpopular. However, some drivers are required to cover the late shifts. Company A wants to maximize the number of satisfied drivers, while company B wants to minimize the total distance between the desired and assigned shifts of all drivers. Figure 1 shows the different functions for selecting alternatives (i.e., ES1, ES2, and MS) to cover late shifts. Company A will select the function on the left to make sure most of the drivers desiring MS get their wish fulfilled, while company B will select the function on the right to gurantee that most of LS is carried by the drivers desiring MS.

Additionally, a linear function as shown in Day and Ryan (1997) is also possible for calculating the penalty costs of alternatives. Due to the limited number of drivers that may simultaneously take a day off, some desired days off (on weekends) can not be satisfied, but can be moved within a couple days earlier or later. These are called moved days off and moved days off on weekends. The number of them are limited due their unpopularity. The number of single-off activities is due to preferences often restricted. The distance between two double-off activities is defined in the company rules to make sure, for example, that a driver get at least two doubles-offs within 16 days.

Real costs: The insufficient assignment of standbys and duties/shifts might cause additional personnel costs or additional costs for overtimes of all planned drivers. The same is for the open days, which should be minimized. Additionally, it should be avoided, for example, that some drivers work overtime while others work substantially less than the regular working time, due to fairness aspect. It can be avoided by minimizing the maximum overtime of all drivers, which results in less payments for overtime.

Fairness: Besides the above mentioned fair distribution of working times, insufficient days off for each driver reflects the fairness of the days off distribution for all drivers. Moreover, the distribution of unpopular activities are embedded in the preferences of drivers (i.e., the desirability of an activity on one particular day).

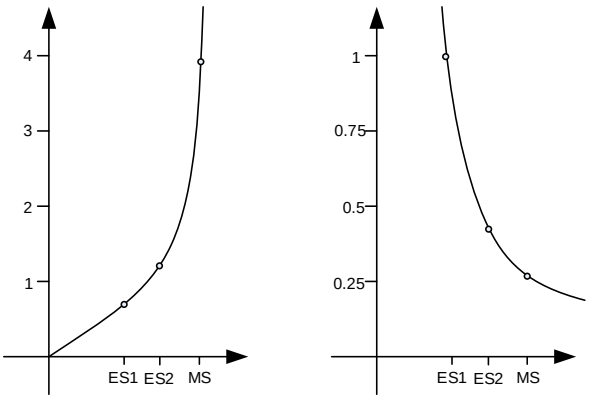

Figure 1 An example of different functions of the penalty costs for the alternatives of late shifts.

Robustness: Usually in operational days many disruptions occur, such as illness. Therefore, a set of standbys should be planned to cover the absences, but the number of them should be minimized to avoid unused standbys. In this work, the number of standbys per day are defined by the bus company by experience. One publication about the optimal planning of the assignment of standbys based on the historical statistics of illness can be found in Xie et al. (2012b).

Moreover, we need to consider the previous planning period, which is assumed to be fixed for the actual planning period. Some other fixed activities are also considered in the current planning periods, such as a training period and annual leaves. We assume that all fixed activities comply with the horizontal and vertical rules. However, between the fixed and unfixed activities the horizontal rules should be checked. The details will be shown in Section 4 together with the example in Figure 3 of the next section.

2.2. Cyclic and non-cyclic crew rostering

There are several ways to generate a roster. A *cyclic roster* is generated for a group of drivers who have the same qualifications and similar preferences. Such a roster includes several rows of duties from Monday to Sunday. The number of the rows is equal to the number of drivers in this group. All drivers within a group use the same roster but begin with different rows. An example is shown in Figure 2, in which different activities, ES (early shift), MS (midday shift), LS (late shift) as well as F (day off), are assigned to two drivers.

In cyclic rostering, the number of days in the planning period is equal to the number of drivers multiplied by seven. Each week is assigned to each driver in such a way that each weekly pattern is worked in parallel with a person. Cyclic rostering is a rather simple way to generate rosters,

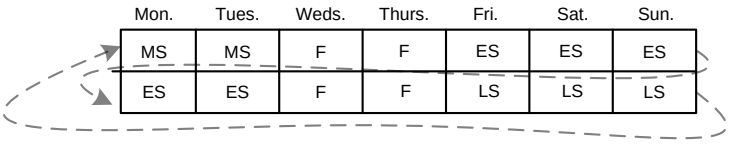

Figure 2 An example of a cyclic roster.

because instead of generating a roster for each driver a roster is generated for a group of drivers. Furthermore, the roster is fair since drivers within a group have the same duties, including unpopular duties, and the days off and weekends-off are evenly distributed. However, a cyclic roster considers the duties from the weekdays, not the calendar days. Therefore, it is not flexible enough to respond to changes in traffic, such as more traffic on holidays. Moreover, a new employee can not be easily added to an existing group, since the planning horizon will be changed, an one more week must be planned.

The shortcomings of cyclic rostering can be avoided if an individual roster is generated for each driver within a given time period (e.g., one month). Such a *non-cyclic roster* may consider wishes with respect to a special single day off or vacation periods. Non-cyclic rostering provides more freedom to take holidays and special events into account. Figure 3 shows an example of a two-week, non-cyclic rosters for two drivers, in which the first five days are in the previous period. As mentioned before, the horizontal rules between the fixed and unfixed activities must be enforced. We assume that the maximum consecutive number of working days is restricted to 5. The five fixed activities in the last period of driver $d2$ are work-related activities. Therefore, a day off activity should be assigned to the first day of the planning period (day 6 in the example). Additionally, we assume the maximum length of a days off period is three days. A working day is therefore required for $d1$ on day 6. We assume that the distance of two double-off activities from the previous period to the current one is defined to be 7. The last double-off in the previous period of $d1$ is on 5th day, and a double-off is required within the next 7 days, such as on days 10 and 11. Such restrictions can be used to reduce the complexity of our network design in Xie and Suhl (2013). Note that the previous period is implicitly considered in the cyclic roster, if the roster is not disrupted.

The non-cyclic rostering approach is widely used in the airline industry, especially in Europe (see Kohl and Karisch (2004), Cappanera and Gallo (2004) and Maenhout and Vanhoucke (2010)). In recent years, this way of generating rosters was also investigated in railways (see De Pont (2006) and Hanne et al. (2009)). In *bidline rostering*, which is used mostly at North American airlines, a roster is generated subsequently for each individual crew member in decreasing order of seniority

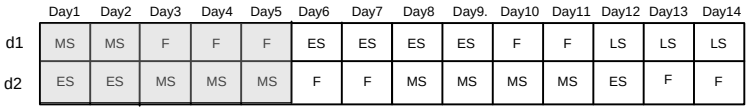

Figure 3 An example of a non-cyclic roster.

in the crew rank (see Kohl and Karisch (2004) for an overview). In the rest of this paper, the cyclic crew rostering is called CCR and non-cyclic crew rostering NCCR.

The crew rostering problem in transportation is differed from the ones in other applications, since it deals with trips/flights. The rostering of call centers, police services, airport ground staffs and general employees deal with flexible demand, while the rostering of nurses and ambulance services deal with shift based demand. More details can be found in Ernst et al. (2004).

The CCR and the NCCR are often solved as a sequence of sub-problems due to their high complexity as shown in Moz and Pato (2007) for the nurse rostering problem. A roster is a schedule of combinations of work-related activities (standby, shift/duty) and day off activities (single-off, double-off). Therefore, the following sub-problems are mostly applied in the literature (see Xie and Suhl (2013) for an overview).

- *days off scheduling*, which is used to assign days off optimally to drivers/driver groups by considering fair distribution of days off.
- *shift assignment*, in which shifts are assigned to drivers based on the given distribution of days off.
- *duty sequencing*, in which duties are assigned to drivers/driver groups according to the result of shift assignment.

Days off scheduling and shift assignment are solved within a step in Emden-Weinert et al. (2000) and Xie et al. (2012b), called the *rota scheduling problem*. Another integration can be made between shift assignment and duty sequencing (called *shift-duty assignment*), i.e., crew rosters are generated based on days off pattern, such as in Townsend (1986), Bianco et al. (1992), Xie et al. (2012a). We solve in this paper these sub-problems within one step, called *integrated planning*.

3. Relevant Research for Crew Rostering

In terms of the mathematical models, the following two relevant models are mostly applied to solve crew rostering in transportation sectors, including public bus transit, airline, rail.

First, the crew rostering problem can be formulated as *set partitioning/covering problem* where each duty is defined as a row and each valid roster is represented as a column. It has been applied widely in air or rail context, such as in Ryan (1992), Day and Ryan (1997), Gamache and Soumis

(1998), Gamache et al. (1999), Freling et al. (2004), De Pont (2006), Medard and Sawhney (2007), Maenhout and Vanhoucke (2010). However, such model is not popular in crew rostering in public bus transit, except for two applications for solving sub-problems, the duty sequencing in Catanas and Paixão (1995) and the days off scheduling in Pedrosa and Constantino (2001). Moreover, column generation is applied for solving a simplified NCCR problem in public transit in Yunes et al. (2005). One reason for that is the size of the real-world instances for the crew rostering problem in public transit is larger than those for the air/rail rostering problem. It is common that a driver can drive different types of vehicles, therefore, many drivers are capable to the same duty that makes the assignment problem more complex. Moreover, unlike a daily duty in public transit, a tour of duties (known as pairing) in the air/rail rostering problem can span up to several days. This creates a large number of duties considered in rosters in public bus transit. Another reason for the fewer applications of column generation for solving crew rostering in public bus transit is that the complicated horizontal rules (including quality rules) make the generation of a feasible and good roster more difficultly (see Section 2.1).

Second, the crew rostering problem can be cast as a *network based mathematical model*. A multi-commodity network flow problem is modeled in Cappanera and Gallo (2004), Xie and Suhl (2013) and Respício et al. (2007) for solving the NCCR, in each network layer activity arcs/nodes and connecting arcs are used to represent the activities and the compatible activities for an employee. For solving the CCR problem, different single-commodity networks were developed. In the network of Caprara et al. (1997), Caprara et al. (1998) and Caprara et al. (1999), nodes represent duties while arcs represent different ways of sequencing the corresponding duties, i.e., directly or with a rest in between. The network in Sodhi and Norris (2004) is applied to solve the rota scheduling problem, where shift patterns (a sequence of shifts without other activities) are represented by nodes and the possible connections between them are given by arcs. Based on the assigned days off pattern, a duty-block network in Xie et al. (2012a) is developed to solve shift-duty assignment problem, where activity arcs/nodes and connecting arcs are used to represent the activities and the compatible activities.

Most of them are solved with heuristics. *Metaheuristics* are widely used to solve multi-objective rostering problem in Monfroglio (1996), Ernst et al. (1998), Lučić and Teodorović (1999) Hanne et al. (2009), El Moudani et al. (2001), Lee and Chen (2003), Lučić and Teodorović (2007), Maenhout and Vanhoucke (2010), Moz et al. (2009), Respício et al. (2007), Xie et al. (2013). Besides that, a *column generation* approach is applied for solving set partitioning/covering-based rostering models in Gamache and Soumis (1998), Gamache et al. (1999), Medard and Sawhney (2007), Catanas and Paixão (1995), Pedrosa and Constantino (2001), where the linear relaxation of the set partitioning problem is solved in the master problem, while in sub-problem (also called as

pricing problem) new columns with negative reduced costs are found. The new columns are added to the master problem to update reduced costs. The sub-problem is formulated as a (constrained) shortest path problem based on a network in which nodes represent pairings/duties, and arcs represent possible links between a pairing and the other activities for each employee. Additionally, the combination of constraint programming and column generation for the air crew rostering problem is discussed in Sellmann et al. (2000). A similar approach is applied for solving the urban transit NCCR problem in Yunes et al. (2005).

The other solution approaches include a constructive heuristic based on *lagrangean relaxation* in Caprara et al. (1997) and Caprara et al. (1999). A linear relaxation is solved with an efficient *branching* strategy in Ryan (1992). And *goal programming* is applied for the multi-objective days off problem in Prakash et al. (1984), and for the multi-objective rostering problem in Respício et al. (2007) and Mesquita et al. (2011).

The main differences between column generation implemented in this paper and the existing one in Yunes et al. (2005).

- We formulate the crew rostering problem as a multi-commodity flow network problem instead of a classic set-partitioning problem.
- We solve both CCR and NCCR problems with same column generation approach.
- We consider a more complex crew rostering problem (see rules in Section 2.1), including components like multiple objectives, balanced workloads, feasibility checks from the previous period to the current one, fixed activities in the current planning period, personal requirements (not only the need for days off, but also considering the daily desired activities including a day off on each calendar day, etc.).
- We generate in each pricing problem a roster with the smallest negative reduce cost. The problem is formulated as an integer program instead of as a constraint satisfaction problem.

The subproblems as mentioned in previous section are often formulated as mixed-integer linear programming problems, which can be solved with a commercial solver or with heuristics. The overview of the corresponding publications can be found in Xie and Suhl (2013).

4. Multi-commodity network design

The NCCR problem in this work is formulated as a multicommodity flow network as in Xie and Suhl (2013). It is extended from the network design for the air crew rotering in Cappanera and Gallo (2004). For each driver, a network layer is generated, where two types of arcs exist: activity and connection arcs. Activity arcs represent valid activities of the driver for each day and connection arcs represent compatible connections between activity arcs. *Compatibility* between two activity arcs means that the direct sequencing of both activities does not violate any horizontal

compatibility rules. Each network layer is acyclic, and a start and end node are defined for each layer. Finding a path in a network layer from the start node to the end node corresponds to a schedule for the driver over a month or several months. On each network layer, the cost of each activity is defined to reflect the desire of the driver, while the cost of each connection arc is used to reflect the desire of sequencing activities of the driver. An example for the desire of sequencing activities for a driver is that similar shifts are connected. Less desired activity arcs and combination arcs receive higher cost. More details about this penalty cost are described in Section 2.1. An illustration of the reduced network of driver d1 and d2 for the example in Figure 3 is shown in Figure 4, in which the different types of activity arcs are illustrated as nodes for a better representation. The network reduction techniques are are discussed in Xie and Suhl (2013). The nodes with letter E, M, L represent shift types early, midday, and late, respectively. The ones with number 1 and 2 represent the single-off and double-off respectively. The marked nodes are the activities selected for the drivers.

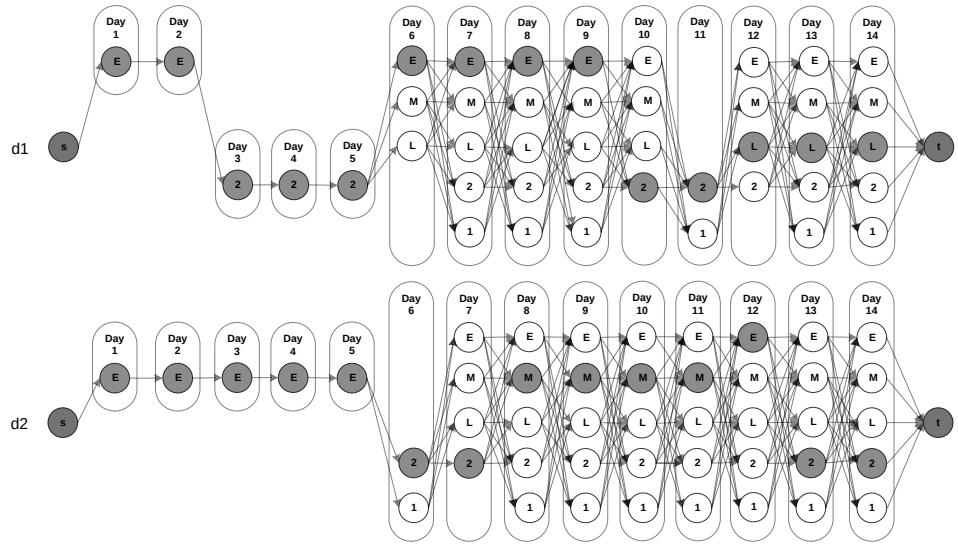

Figure 4 Illustration of the network layers of the exmaple in Figure 3.

The same network design and reducing techniques for NCCR are used for solving the CCR problem. However, as in Xie and Suhl (2013), the following differences should be noticed during the formulation of models.

- We consider each driver group as one driver.
- For each driver there might be a different planning period, which is calculated as the number of drivers in the driver group multiplied by the number of weekdays within a week (i.e, seven).

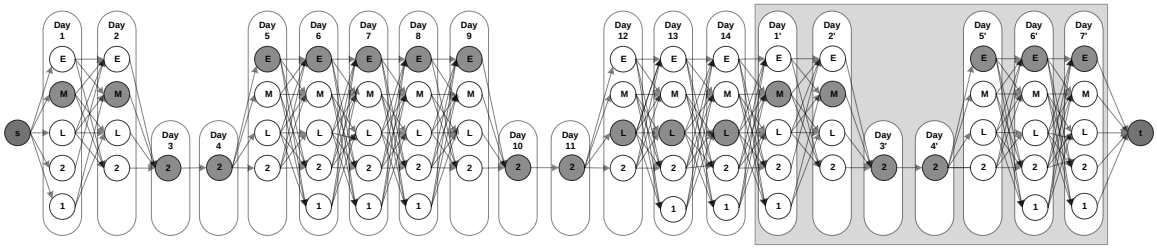

Figure 5 Illustration of the network layer of the CCR problem in Figure 2.

- The capacity of each work-related activity is no longer restricted by days, but weekdays.
- Due to the property of CCR, the compatibility of the assigned activity on the last day with the one on the first day should be checked. Additionally, the length of working, days off blocks as well as the length of maximal distance between two double-off arcs should be checked. In order to retain the acyclic network, a sequence of artificial days is appended to the planning period whose length is equal to the largest block length we consider (such as the maximal distance between two double-off arcs). The available activity arcs of these artificial days are equal to those at the beginning of the planning period.

The network for the example shown in Figure 2 is illustrated in Figure 5. One network layer is generated for two drivers within a group in the example. Additionally, the maximal distance between two double-off arcs is defined as seven. Therefore, seven days are append to the end of the planning horizon, which in this example is 14 days.

5. Column Generation for the NCCR

In this section, the column generation approach for the NCCR is described, which includes the generation of an initial solution in Section 5.1, and the description of the master and pricing problem models (see Section 5.2 and 5.3). A short description of sub-problems is shown in Section 5.4.

We first begin with a definition of basic and specialized index-sets, all necessary parameters and variables.

Index-sets

$\mathcal{M}$	Set of all employees.
$\mathcal{T}_m$	Partially ordered set of all discrete time-elements t (here: days) of the planning horizon, which means $\mathcal{T}_m = \{1, \ldots, T_m\}\ \forall m \in \mathcal{M}$.
$\mathcal{D}$	Set of all activities d.
$\mathcal{D}_t \subseteq \mathcal{D}$	Set of all available activities on day $t \in \mathcal{T}_m$.
$\mathcal{D}_w \subseteq \mathcal{D}$	Set of all work-related activities.
$\mathcal{D}_b \subseteq \mathcal{D}_w$	Set of all stand by activities.
$\mathcal{D}_s \subseteq \mathcal{D}_w$	Set of all shift activities.
$\mathcal{D}_f \subseteq \mathcal{D}$	Set of all day off activities.

$\mathcal{D}_{df} \subseteq \mathcal{D}_f$	Set of all double-off activities.
$\mathcal{D}_{sf} \subseteq \mathcal{D}_f$	Set of all single-off activities.
$\mathcal{D}_{mf} \subseteq \mathcal{D}_f$	Set of all moved day off activities.
$\mathcal{D}_{mfw} \subseteq \mathcal{D}_f$	Set of all moved day off activities at a weekend.
$\mathcal{D}_{du} \subseteq \mathcal{D}$	Set of all dummy activities.
$\mathcal{J}$	Set of duties.
$\mathcal{J}_e$	Set of duties belonging to the activity arc e, where $e \in \mathcal{A}_{m,t,d}, d \in \mathcal{D}_s, t = T_d, m \in \mathcal{M}$.
$\mathcal{E} = \mathcal{A} \cup \mathcal{C}$	Set of all edges e. This set include the set of all activity arcs $\mathcal{A}$ implied by the given activities with the set of all compatibility arcs $\mathcal{C}$.
$\mathcal{A} \subset \mathcal{E}$	Set of all activity arcs.
$\mathcal{A}^s \subset \mathcal{E}$	Set of all activity arcs e including the extended source e^s_m and sink-edges e^e_m per employee $m \in \mathcal{M}$. That is, $\mathcal{A}^s = \bigcup_{m \in \mathcal{M}} \{e^s_m, e^e_m\} \cup \mathcal{A}$
$\mathcal{A}^s_m \subseteq \mathcal{A}^s$	Subset of $\mathcal{A}^s$ that contains all activity arcs including the extended source e^s_m and sink-edges e^e_m of the employee $m \in \mathcal{M}$.
$\mathcal{A}_{m,t,d} \subseteq \mathcal{A}$	Subset of $\mathcal{A}$ that only contains the activities available to a given employee $m \in \mathcal{M}$ on day $t \in \mathcal{T}_m$ referring to an activity $d \in \mathcal{D}$.
$\mathcal{A}_j$	Set of possible shift activity arcs, to which the duty $j \in \mathcal{J}$ can be assigned.
$\mathcal{A}^a$	Set of all shift activity arcs.
$\mathcal{A}^a_m$	Set of all shift activity arcs of driver $m \in \mathcal{M}$.
$\mathcal{C} \subset \mathcal{E}$	Set of all compatibility arcs e, which connect compatible activity-arcs.
$\mathcal{F}_e \subset \mathcal{C}$	Set of all compatibility-arcs outgoing from activity-arc e, which link to all compatible activity arcs of the following day for a specific employee. The so-called "Forward-Star" (see also Cappanera and Gallo (2004)), with $e \in \mathcal{A}^s$.
$\mathcal{B}_e \subset \mathcal{C}$	Set of all compatibility arcs incoming to activity arc e, which link to all compatible activity arcs of the preceding day for a specific employee. The so-called "Backward-Star" (see also Cappanera and Gallo (2004)) with $e \in \mathcal{A}^s$.
$\mathcal{O} \subseteq \mathcal{C}$	Contains all compatibility arcs e that are forbidden due to incompatibility.
$\mathcal{S}$	Contains all sets s of sequencing activity arcs, which are forbidden to be simultaneously active in a feasible solution. This means, for example, $(e_i, \ldots, e_j) \in \mathcal{S}$ with $e_i, \ldots, e_j \in \mathcal{A}$. These sets of edges are implied by the given forbidden sequences. Due to the reduction in the network, each $\lvert s \rvert > 2$.
$\mathcal{S}_m \subset \mathcal{S}$	Set s of forbidden sequencing activity arcs for the driver m.
$\mathcal{S}^o_m$	Contains all sets s of sequencing duties (length 2) for the driver m, which are forbidden to be simultaneously active in a feasible solution due to violating the overnight rest period.
$\mathcal{X}$	Set of all activity and compatibility arcs as well as duties of a feasible solution.

Parameters

T_m	Count of days of the planning horizon for employee m.
D	Maximal distance between two double-off arcs.
H	Minimal rest hours between two work-related activities.
L^R_m	Maximal length of consecutive day off activities for employee m.
L^W_m	Maximal length of consecutive work-related activities for employee m.

C^d	Penalty cost for not satisfying the maximal distance, defined by D, between two double-off activities.
C^f	Penalty cost for falling short of the minimal amount of day off activities per employee.
C^g	Penalty cost for exceeding the maximal amount of single-off activities per employee.
C^p	Penalty cost for falling below regarding the minimal count of standbys of an employee.
C^u	Penalty cost for an unallocated shift/duty.
C^{m1}	Penalty cost for activating an activity which indicates a moved day off activity.
C^{m2}	Penalty cost for activating an activity which indicates a moved day off activity at a weekend.
C^o	Penalty cost for the maximum overtime of all drivers.
C_e^a	Cost which appear when using the activity e.
C^{du}	Penalty cost for an unassigned day per employee.
C_e^r	Reduced cost of selecting the activity arc $e \in \mathcal{A}_m^s$ for the employee $m \in \mathcal{M}$.
C_j^r	Reduced cost of selecting the duty $j \in \mathcal{J}_e$ of the activity arc $e \in \mathcal{A}_m^s$ for the employee $m \in \mathcal{M}$.
I_d	Count of units of an activity $d \in \mathcal{D}_w \cup \mathcal{D}_f$ which are planned to get allocated.
W_d	Work time consumed when using an activity belonging to the work-related activity $d \in \mathcal{D}_w$.
W_j	Work time consumed when using a duty $j \in \mathcal{J}$.
T_d	The day of the work-related activity $d \in \mathcal{D}_w$.
F_m	Minimal count of days off for employee m.
E_m	Maximal count of single-off activities for employee m.
V_m	Minimal count of standbys for employee m.
R_m	Desired amount of working hours for employee m. When exceeding this value for the correlating employee the resulting surplus will be considered overtime.
A_e	Outgoing activity arc of the compatibility arc $e \in \mathcal{C}$.

Variables

x_e^a	Binary variable that depicts the flow over the activity arc $e \in \mathcal{A}^s$.
x_e^c	Binary variable, which depicts the flow over the compatibility arc $e \in \mathcal{C}$.
e_m^s	Binary variable for defining the source edge of employee $m \in \mathcal{M}$.
e_m^e	Binary variable for defining the sink edge of employee $m \in \mathcal{M}$.
$y_{e,j}$	Binary variable that is equal to one if the duty j is assigned to the activity arc $e \in \mathcal{A}_{m,t,d}$, and 0 otherwise.
σ_m	Sum of the days off for the employee m described through an integer-variable with the bounds $l = 0$ and $u = T^m$.
v_m	Binary variable that indicates if the sum of days off is satisfactory for the corresponding employee m.
τ_m	Sum of single-off activities for the employee m depicted by an integer variable with the bounds $l = 0$ and $u = T^m$.
w_m	Binary variable that indicates if the sum of single-off activities is overflown for the employee m.
ψ_m	Sum of the standbys of the employee m. This integer variable is bounded between $l = 0$ and $u = T^m$.

p_m	Binary variable, which indicates if the sum of used standbys is falling short for employee m.
λ_d	Sum of not allocated units of activity d depicted by an integer variable within the bounds $l = 0$ and $u = I_d$.
d_m	Binary variable, which indicates if a violation of the maximal distance between two double-off activities is present for the employee m.
ϕ	Maximal overtime over all employees as a positive continuous variable.
z_m	Integer variable that indicate the sum of open days for the employee m, since it is possible that there are some days without assigned activities for the employee m.

5.1. Initial Solution

The initial solution $\mathcal{X}$ can be achieved by a construction heuristic driver-by-driver with back-tracking (see Algorithms 1). Let $\mathcal{X}$ be a set of vectors (activity, duty). The horizontal and vertical rules are considered in the construction heuristic, i.e., the generated solution for each driver is feasible. In $\mathcal{F}'_n$ the compatibility arcs e are collected such that adding their corresponding activity arcs e' to the current solution does not violate any maximum block lengths or forbidden sequences. An examples about detecting forbidden sequences is illustrated in Figures 6. Moreover, the capacity of each activity should be controlled during the generation. Therefore, the method we generate a initial solution is called as a look-ahead method. If it exists no valid choice, then back-tracking (Algorithm 2) is activated until a valid choice is available or a source-node is reached.

In the example shown in Figure 6, three forbidden sequences are considered, which are illustrated as branches of trees. An early shift activity is represented by a node with mark E/0, while M/0 and L/0 mean a midday and late shift activity, respectively. A standby activity is marked by 0/4. A duty $j \in \mathcal{J}_{e'}$ is then feasible if j is compatible with the previous duty (i.e., it does not violate overnight rest time) and it is available. A check of validity of adding an activity can be done by checking its corresponding node in the tree up to the root nodes and down to all leafs, if necessary. The reference between each activity and itself in each forbidden tree is attached during the generation of a solution. Each reference is illustrated by a dotted line. In this example, by adding the early shift (E/0) to the actual solution, this shift is attached to the leaf of the tree (by a dotted line), however, this early shift is forbidden to be added to the actual solution. Since the previous two days late shift and midday shift are selected sequentially (the forbidden sequence (LS, MS, ES)).

In the function UpdateMetaInfo() of lines 15 and 26 in Algorithm 1, the actual block length of working days and days off as well as the capacity of an activity are updated. An example illustrated in Figure 7 shows after the checking of feasibility, block length of working days and days off are updated. The capacity of each activity is updated for all drivers on each day (for the NCCR)/week day (for the CCR).

Algorithm 1 The construction algorithm for generating a feasible initial solution.

procedure DRIVER-BY-DRIVER
  $\mathcal{X} \leftarrow \emptyset$
  for all driver $m \in \mathcal{M}$ **do**
    $n \leftarrow e^s_m$
    while $n \neq$ nil **do**
      $\mathcal{F}'_n \leftarrow$ Subset of randomly sorted feasible outbound compatibility arcs $e \in \mathcal{F}_n$
      if $\mathcal{F}'_n = \emptyset$ **then**
        $n \leftarrow$ BACKTRACKREPAIR
      end if
      $e \leftarrow Pop(\mathcal{F}'_n)$
      while $e \neq nil$ **do**
        $e' \leftarrow A_e$
        if $\mathcal{J}_{e'} \neq \emptyset$ **then**
          if $\exists$ a feasible duty $j \in \mathcal{J}_{e'}$ **then**
            $\mathcal{X} \leftarrow \mathcal{X} \cup \{(e', j)\}$, UpdateMetaInfo()
            $n \leftarrow e'$
            break
          else
            $e \leftarrow Pop(\mathcal{F}'_n)$
            if $e == nil$ **then**
              $n \leftarrow$ BACKTRACKREPAIR
              break
            end if
          end if
        else
          $\mathcal{X} \leftarrow \mathcal{X} \cup \{(e', nil)\}$, UpdateMetaInfo()
          $n \leftarrow e'$
          break
        end if
      end while
    end while
  end for
  return $\mathcal{X}$
end procedure

Algorithm 2 The backtracking procedure used for the construction algorithm

procedure BACKTRACKREPAIR

$k \leftarrow nil$

do

if $k \neq nil$ **then**

$\mathcal{K} \leftarrow \mathcal{K} \setminus \{k\}$

end if

$k \leftarrow Pop(\mathcal{X})[0]$

UpdateMetaInfo()

$\mathcal{K} \leftarrow \mathcal{K} \cup k$

while $\mathcal{F}'_k = \emptyset$

return k

end procedure

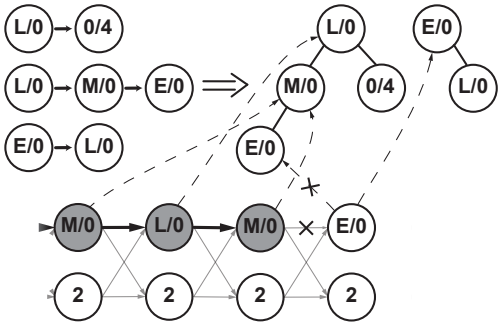

Figure 6 **Tracking forbidden sequences.**

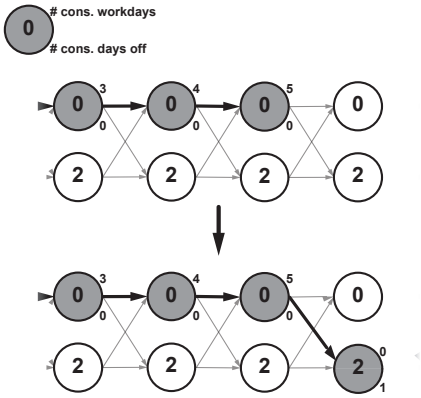

Figure 7 **Update of block length.**

In the back-tracking algorithm, $\mathcal{K}$ is defined as a set of activity arcs, similar to a tabu list. The algorithm gives activity arc k as output, which indicates the closest possible activity to backtrack. From that activity begins the search of the rest of initial solution.

It is worth to mention that, Algorithm 1 is suitable for both CCR and NCCR problems, as well as the rota scheduling problem of them.

5.2. The Master Problem

All quality rules described in Section 2.1 are considered in the master problem. Those rules are formulated as soft constraints and can be weighted by applying specific penalty costs to them.

Preferences The costs of selected undesired activities in objective (1), and the costs for moved days off (at weekends) in objective (2) and (3). Additionally, another cost reflects the desirability of the frequency of a desired/undesired activity, which contains the costs of underrun days off. The fulfillment of the minimal sum of days off per driver is achieved by the constraint set (14). The first group of constraints sums the count of days off per driver, while the second group determines whether the minimal sum of days off is met. If it is not met for a driver, then the penalty cost in objective (4) is activated. The same is for overrun single-off activities (see constraint set (15) and objective (5)). Moreover, constraint set (16) checks weather at least one violation of the maximal distance between two double-off activities is occurred for each driver. In that case, objective (6) is activated.

Real costs are considered as follows. The number of unassigned shifts is counted as λ_d in the constraint set (13) and minimized in objective (8). Additionally, if a given number of standbys falls short for a driver, the corresponding penalty costs will be added to the objective function (see constraint set (17) and objective (7)). In order to avoid the case that there is no activity possible for a driver on a day, a dummy activity is selected. In (18), the sum of all open days for a driver m is stored in the integer variable z_m. And the number of them are minimized in objective (9). Moreover, in order to prevent the large differences of overtimes among drivers, the constraint set (20) defines the maximal overtime of all drivers, which is minimized in objective (10).

Fairness An even distribution of workload (overtime) is considered in constraints (20) and objective (10). The other one, i.e., the given number of days off for each driver, should be satisfied as far as possible (see constraint set (14) and objective (4)).

Constraints As described above, the NCCR is formulated as a multi-commodity network flow problem, which contains the following *network constraints*. The set of constraints (11) makes sure that the value of an activity arc matches the sum of all compatibility arcs of the backward-star as well as the sum of all compatibility arcs of the forward-star (*flow conservation*), while the set of constraints (12) ensures that exactly one activity is selected for the driver m per day (*demand satisfaction*). Additionally, the *capacity constraints* (19) and (21) in the vertical rules are considered in order to check the compatibility of the generated rosters of all drivers. Moreover, the relation between each shift activity arc and its corresponding duties is taken into account (see constraint

set (22)), i.e., each assigned shift gets a corresponding duty. We do not consider all variables in the master problem, but only some of them. Therefore, this problem is called the *restricted master problem*. The considered variables are relaxed, i.e., they are linear variables.

$$
\begin{aligned}
&\min z^M_{NCCR} \\
&= \sum_{e \in \mathcal{A}^s} C^a_e \cdot x^a_e && \text{(Costs of selected activities)} && (1)\\
&+ \sum_{m \in \mathcal{M}} \sum_{d \in \mathcal{D}_{mf}} \sum_{e \in \mathcal{A}_{m,T_d,d}} C^{m1} \cdot x^a_e && \text{(Moved days off)} && (2)\\
&+ \sum_{m \in \mathcal{M}} \sum_{d \in \mathcal{D}_{mfw}} \sum_{e \in \mathcal{A}_{m,T_d,d}} C^{m2} \cdot x^a_e && \text{(Moved days off at a weekend)} && (3)\\
&+ \sum_{m \in \mathcal{M}} C^f \cdot v_m && \text{(Costs of under run days off)} && (4)\\
&+ \sum_{m \in \mathcal{M}} C^g \cdot w_m && \text{(Costs for overflown single-offs)} && (5)\\
&+ \sum_{m \in \mathcal{M}} C^d \cdot \delta_m && \text{(Violated dist. bet. double-offs)} && (6)\\
&+ \sum_{m \in \mathcal{M}} C^p \cdot p_m && \text{(Underrun standbys)} && (7)\\
&+ \sum_{d \in \mathcal{D}_s} C^u \cdot \lambda_d && \text{(Costs of unassigned shifts)} && (8)\\
&+ \sum_{m \in \mathcal{M}} C^{du} \cdot z_m && \text{(Costs of open days)} && (9)\\
&+ C^o \cdot \phi && \text{(Overtime)} && (10)
\end{aligned}
$$

s.t.

$$
\begin{aligned}
x^a_e &= \sum_{e' \in \mathcal{F}_e} x^c_{e'} && \forall e \in \mathcal{A}^s && (11)\\
x^a_e &= \sum_{e' \in \mathcal{B}_e} x^c_{e'} && \forall e \in \mathcal{A}^s \\
e^s_m &= 1 && \forall m \in \mathcal{M} && (12)\\
e^e_m &= 1 && \forall m \in \mathcal{M} \\
I_d - \sum_{m \in \mathcal{M}} \sum_{e \in \mathcal{A}_{m,t,d}} x^a_e &= \lambda_d && \forall d \in \mathcal{D}_s, t = T_d && (13)\\
\sum_{e \in \mathcal{A}_{m,t,d}} x^a_e &= \sigma_m && \forall m \in \mathcal{M}, t \in \mathcal{T}_m, d \in \mathcal{D}_f && (14)\\
F_m - v_m \cdot T^m &\leq \sigma_m && \forall m \in \mathcal{M} \\
\sum_{e \in \mathcal{A}_{m,t,d}} x^a_e &= \tau_m && \forall m \in \mathcal{M}, t \in \mathcal{T}_m, d \in \mathcal{D}_{sf} && (15)
\end{aligned}
$$

$$E_m + w_m \cdot T^m \geq \tau_m \qquad \forall m \in \mathcal{M}$$

$$\sum_{t'=t}^{t+D} \sum_{e \in \mathcal{A}_{m,t',d}} x_e^a \geq 1 - d_m \qquad \forall m \in \mathcal{M}, d \in \mathcal{D}_{df}, t \in \{1, \ldots, T_m - D\} \tag{16}$$

$$\sum_{e \in \mathcal{A}_{m,t,d}} x_e^a = \psi_m \qquad \forall m \in \mathcal{M}, t \in \mathcal{T}_m, d \in \mathcal{D}_b \tag{17}$$

$$\psi_m \geq V_m - p_m \cdot V_m \qquad \forall m \in \mathcal{M}$$

$$\sum_{d \in \mathcal{D}_{du}} \sum_{e \in \mathcal{A}_{m,t,d}} x_e^a = z_m \qquad \forall m \in \mathcal{M} \tag{18}$$

$$\sum_{m \in \mathcal{M}} \sum_{e \in \mathcal{A}_{m,t,d}} x_e^a \leq I_d \qquad \forall d \in \mathcal{D}_w \cup \mathcal{D}_f, t = T_d \tag{19}$$

$$\sum_{d \in \mathcal{D}_w \setminus \mathcal{D}_s} \sum_{e \in \mathcal{A}_{m,T_d,d}} W_d \cdot x_e^a + \tag{20}$$

$$\sum_{d \in \mathcal{D}_d} \sum_{e \in \mathcal{A}_{m,T_d,d}} \sum_{j \in \mathcal{J}_e} W_j \cdot y_{e,j} -$$

$$R_m \leq \phi \qquad \forall m \in \mathcal{M}$$

$$\sum_{e \in \mathcal{A}_j} y_{e,j} \leq 1 \qquad \forall j \in \mathcal{J} \tag{21}$$

$$x_e^a = \sum_{j \in \mathcal{J}_e} y_{e,j} \qquad \forall e \in \mathcal{A}^a \tag{22}$$

5.3. The Pricing Problem

The pricing problem aims generate a feasible roster with minimal reduced costs for each driver in objective (23). Recall that, C_e^r is defined as the reduced cost of selecting the activity arc $e \in \mathcal{A}_m^s$ for the employee $m \in \mathcal{M}$. And C_j^r is the reduced cost of selecting the duty $j \in \mathcal{J}_e$ of the activity arc $e \in \mathcal{A}_m^s$ for the employee $m \in \mathcal{M}$. They are updated after solving the master problem. The following pricing model is run in parallel for each driver $m \in \mathcal{M}$. It is based on the hard constraints of the integrated model for the NCCR in Xie and Suhl (2013), but without the capacity constraints. Moreover, constraints (22a) ensure each assigned shift activity arc get an assigned duty, while constraints (28) forbid the impossible combination of duties.

We consider in each pricing problem only one network layer for a driver. Therefore, the network constraints for a single network layer are considered during the generation of a feasible schedule for a driver. The flow conservation constraints (11a) and the demand satisfaction constraints (12a) are defined for each driver. Additionally, a resource constrained shortest path has to be determinated in each network layer. The resources include the maximal block length of work-related days (see constraint set (24)) as well as the maximal block length of days off for the driver m (see constraint set (25)). Additionally, the double-off as well as forbidden sequences are defined in the constraints (26) and (27), respectively. All variables in the pricing problem are integer variables.

$$\min z^P_{NCCR} = \sum_{d\in\mathcal{D}_s}\sum_{e\in\mathcal{A}_{m,T_d,d}}(C^r_e\cdot x^a_e + \sum_{j\in\mathcal{J}_e} C^r_j\cdot y_{e,j}) + \sum_{d\in\mathcal{D}\setminus\mathcal{D}_s}\sum_{e\in\mathcal{A}_{m,T_d,d}} C^r_e\cdot x^a_e \tag{23}$$

s.t.

$$x^a_e = \sum_{e'\in\mathcal{F}_e} x^c_{e'} \qquad \forall e\in\mathcal{A}^s_m \tag{11a}$$

$$x^a_e = \sum_{e'\in\mathcal{B}_e} x^c_{e'} \qquad \forall e\in\mathcal{A}^s_m$$

$$e^s_m = 1 \tag{12a}$$

$$e^e_m = 1$$

$$\sum_{t'=t}^{t+L^W_m}\sum_{e\in\mathcal{A}_{m,t',d}} x^a_e \geq 1 \qquad t\in\{1,...,T_m - L^W_m\}, d\in\mathcal{D}_f \tag{24}$$

$$\sum_{t'=t}^{t+L^R_m}\sum_{e\in\mathcal{A}_{m,t',d}} x^a_e \geq 1 \qquad t\in\{1,...,T_m - L^R_m\}, d\in\mathcal{D}_w \tag{25}$$

$$\sum_{e\in\mathcal{A}_{m,t,d}} x^a_e \geq \sum_{e\in\mathcal{A}_{m,t+1,d}} x^a_e \qquad t\in\{1,...,T_m-1\}, d\in\mathcal{D}_{df} \tag{26}$$

$$\sum_{e\in s} x^a_e \leq |s| - 1 \qquad \forall s\in\mathcal{S}_m \tag{27}$$

$$y_{e,j} + y_{e',j'} \leq 1 \qquad \forall e\in\mathcal{A}_j, e'\in\mathcal{A}_{j'}, (j,j')\in\mathcal{S}^o_m \tag{28}$$

$$x^a_e = \sum_{j\in\mathcal{J}_e} y_{e,j} \qquad \forall e\in\mathcal{A}^a_m \tag{22a}$$

5.4. Subproblem: Rota Scheduling (RS)

As shown in Xie and Suhl (2013), the rota scheduling problem is hard to solve, therefore, a similar column generation approach is also applied to solve the rota scheduling problem. The mathematical model of its sequencing process, i.e., the duty sequencing problem, can be found in Xie and Suhl (2013).

The Master Problem

All constraints in the master problem of the integrated planning for the NCCR in Section 5.2 are also considered in the model of this problem, except the constraints about duties, (21) and (22). Moreover, the constraint set defining the maximum overtime for all drivers (20) is modified by (20a), and its corresponding objective (10a).

$$\min z^{M,RS}_{NCCR} = (1)+(2)+(3)+(4)+(5)+(6)+(7)+(8)+(9)+C^o\cdot\phi' \tag{10a}$$

s.t.

$$(11)-(19)$$

$$\sum_{d\in\mathcal{D}_w}\sum_{e\in\mathcal{A}_{m,T_d,d}} W_d - R_m \leq \phi' \qquad \forall m\in\mathcal{M} \tag{20a}$$

The Pricing Problem

This problem aims at finding a feasible combination of activities for the driver m with the lowest reduced cost (See objective (23a)). All constraints in the pricing problem of the crew rostering problem in Section 5.3 are also considered in the model of this problem, except the constraints about duties (28) and (22a).

$$\min z_{NCCR}^{P,RS} = \sum_{e\in\mathcal{A}_m^s} C_e^r \cdot x_e^a \qquad \text{(Reduced costs)} \tag{23a}$$

s.t.

$$(11a),(12a),(24),(25),(26),(27)$$

6. Column Generation for the CCR

As mentioned in Section 4, the differences between this problem and the NCCR problem is that the capacity of each work-related activity is no longer restricted by days, but weekdays. Additionally, the constraints about the artificial days should be considered. Therefore, $T_d{}'$ is defined as the weekday of the work-related activity $d\in\mathcal{D}_w$. Note that $\mathcal{T}_m$ is defined as a set of days for driver m, including the artifical days for the CCR problem. $\mathcal{T}_m^{art}\subset\mathcal{T}_m$ is defined as the set of artifical days in the network of driver m. A set $\mathcal{T}_{m,t}$ is used to define the days (but not including artifical days) in the network of the weekday t for the driver $m\in\mathcal{M}$. For example, the weekday t is equal to 1, so a set $\mathcal{T}_{m,t}$ might be $\{1,8,15,...\}$. The length of the artificial days for each driver m is stored in b_m^{art}. Note that we use the similar initial solution as shown in Section 5.1 for the NCCR problem.

6.1. The Master Problem

The artificial days should be ignored by the constraints of calculations of unassigned shifts ($13a$), insufficient days off ($14a$), overrun single-offs ($15a$), insufficient standbys ($17a$), the unassigned days ($18a$), as well as the maximum overtime ($20b$). However, the artificial days are considered in the constraints of flow conservation (11), demand satisfaction (12) as well as checking maximal distance between two double-off activities (16). Constraints ($13a$) and ($19a$) make sure that the work-related activities are restricted by weekdays. Constraints (29) and (30) make sure the selected activity and duty on each artificial day $T_m - b_m^{art} + i$ is equal to the selected ones on the day i, where

$i \in \{1, 2, ..., b_m^{art}\}$. Additionally, the relation between a shift activity arc and its possible duties is guaranteed in (22), and the duty capacity constraints in (21) should be considered as well.

$$
\begin{aligned}
\min z_{CCR}^M = & && (1)+(2)+(3)+(6) && \\
& + \sum_{m \in \mathcal{M}} C^f \cdot {v_m}' && \text{(Costs of underrun days off)} && (4a) \\
& + \sum_{m \in \mathcal{M}} C^g \cdot {w_m}' && \text{(Costs for overflown single-offs)} && (5a) \\
& + \sum_{m \in \mathcal{M}} C^p \cdot {p_m}' && \text{(Underrun standbys)} && (7a) \\
& + \sum_{d \in \mathcal{D}_s} C^u \cdot {\lambda'}_d && \text{(Costs of unassigned shifts)} && (8a) \\
& + \sum_{m \in \mathcal{M}} C^{du} \cdot {z_m}' && \text{(Costs of open days)} && (9a) \\
& + C^o \cdot \phi'' && \text{(Overtime)} && (10b)
\end{aligned}
$$

s.t.

$$
\begin{aligned}
& (11), (12), (16), (21), (22), (28) && && \\
& I_d - \sum_{m \in \mathcal{M}} \sum_{t' \in \mathcal{T}_{m,t}} \sum_{e \in \mathcal{A}_{m,t',d}} x_e^a = {\lambda_d}' && \forall d \in \mathcal{D}_s, t = {T_d}' && (13a) \\
& \sum_{e \in \mathcal{A}_{m,t,d}} x_e^a = {\sigma_m}' && \forall m \in \mathcal{M}, d \in \mathcal{D}_f, t \in \mathcal{T}_m \setminus \mathcal{T}_m^{art} && (14a) \\
& F_m - {v_m}' \cdot T^m \leq {\sigma_m}' && \forall m \in \mathcal{M} && \\
& \sum_{e \in \mathcal{A}_{m,t,d}} x_e^a = {\tau_m}' && \forall m \in \mathcal{M}, d \in \mathcal{D}_{sf}, t \in \mathcal{T}_m \setminus \mathcal{T}_m^{art} && (15a) \\
& E_m + {w_m}' \cdot T^m \geq {\tau_m}' && \forall m \in \mathcal{M} && \\
& \sum_{e \in \mathcal{A}_{m,t,d}} x_e^a = {\psi_m}' && \forall m \in \mathcal{M}, d \in \mathcal{D}_b, t \in \mathcal{T}_m \setminus \mathcal{T}_m^{art} && (17a) \\
& {\psi_m}' \geq V_m - {p_m}' \cdot V_m && \forall m \in \mathcal{M} && \\
& \sum_{t \in \mathcal{T}_m \setminus \mathcal{T}_m^{art}} \sum_{d \in \mathcal{D}_{du}} \sum_{e \in \mathcal{A}_{m,t,d}} x_e^a = {z_m}' && \forall m \in \mathcal{M} && (18a) \\
& \sum_{m \in \mathcal{M}} \sum_{t' \in \mathcal{T}_{m,t}} \sum_{e \in \mathcal{A}_{m,t',d}} x_e^a \leq I_d && \forall d \in \mathcal{D}_w, t = {T_d}' && (19a) \\
& \sum_{t \in \mathcal{T}_m \setminus \mathcal{T}_m^{art}} \sum_{d \in \mathcal{D}_{w,t} \setminus \mathcal{D}_{s,t}} \sum_{e \in \mathcal{A}_{m,t,d}} W_d \cdot x_e^a + && && (20b) \\
& \sum_{t \in \mathcal{T}_m \setminus \mathcal{T}_m^{art}} \sum_{d \in \mathcal{D}_{s,t}} \sum_{e \in \mathcal{A}_{m,t,d}} \sum_{j \in \mathcal{J}'_e} W_j \cdot y_{e,j} - && && \\
& R_m \leq \phi'' && \forall m \in \mathcal{M} && \\
& x_e^a = x_{e'}^a && \forall m \in \mathcal{M}, i \in \{1, 2, ..., b_m^{art}\}, d \in \mathcal{D}_i, && (29)
\end{aligned}
$$

$$
\begin{aligned}
& & & e \in \mathcal{A}_{m,T_m-b_m^{art}+i,d}, e' \in \mathcal{A}_{m,i,d} \\
& y_{e,j} = y_{e',j} & & \forall m \in \mathcal{M}, i \in \{1,2,...,b_m^{art}\}, e, e' \in \mathcal{A}_j, \\
& & & T_e = i, T_{e'} = T_m - b_m^{art} + i, j \in \mathcal{J}_e
\end{aligned} \tag{30}
$$

6.2. The Pricing Problem

We extend the pricing problem for the NCCR problem (constraints (11a) to (22a)) with constraints (29a) and (30a) to make sure the selected activity and duty on each artificial day are equal to the selected ones on the original day for the driver m. Additionally, it is known that the duties are weekday dependent, therefore, we have to make sure that each duty can only assigned to a driver once. We assume that $\mathcal{A}_j$ does not contain shift activities on artificial days. Therefore, the constraints (21) from the Section 5.3 remain.

$$
\begin{aligned}
\min z_{CCR}^P = & \sum_{t \in \mathcal{T}_m \setminus \mathcal{T}_m^{art}} \sum_{d \in \mathcal{D}_{s,t}} \sum_{e \in \mathcal{A}_{m,t,d}} (C_e^r \cdot x_e^a + \sum_{j \in \mathcal{J}_e} C_j^r \cdot y_{e,j}) \qquad \text{(Reduced costs)} \\
& + \sum_{t \in \mathcal{T}_m \setminus \mathcal{T}_m^{art}} \sum_{d \in \mathcal{D}_{w,t} \setminus \mathcal{D}_{s,t}} \sum_{e \in \mathcal{A}_{m,t,d}} C_e^r \cdot x_e^a
\end{aligned} \tag{23b}
$$

s.t.

$$
(11a), (12a), (24), (25), (26), (27), (28), (22a), (21)
$$

$$
x_e^a = x_{e'}^a \qquad i \in \{1,2,...,b_m^{art}\}, d \in \mathcal{D}_i, \tag{29a}
$$

$$
\begin{aligned}
& & & e \in \mathcal{A}_{m,T_m-b_m^{art}+i,d}, e' \in \mathcal{A}_{m,i,d} \\
& y_{e,j} = y_{e',j} & & i \in \{1,2,...,b_m^{art}\}, e, e' \in \mathcal{A}_m, \\
& & & T_e = i, T_{e'} = T_m - b_m^{art} + i, j \in \mathcal{J}_e
\end{aligned} \tag{30a}
$$

6.3. Subproblem: Rota Scheduling (RS)

It is similar to Section 5.4, all constraints in the master and pricing problem of the CCR in Section 6.1 and 6.2 are considered in the models of this section, except the constraints about duties in the master problem (21), (22), (28), and (30) as well as the constraints in the pricing problem (21), (22a), (28),and (30a).

The Master Problem

The constraint set about the maximum overtime is replaced with $(20c)$.

$$
\begin{aligned}
\min z_{CCR}^{M,RS} & \\
& = (1) + (2) + (3) + (4a) + (5a) + (6) + (7a) + (8a) + (9a) \\
& + C^o \cdot \phi''' \qquad \text{(Overtime)}
\end{aligned} \tag{10c}
$$

s.t.

$$(11), (12), (13a), (14a), (15a), (16), (17a), (18a), (19a), (29)$$

$$\sum_{t \in \mathcal{T}_m \setminus \mathcal{T}_m^{art}} \sum_{d \in \mathcal{D}_w} \sum_{e \in \mathcal{A}_{m,t,d}} W_d \cdot x_e^a - R_m \leq \phi''' \qquad \forall m \in \mathcal{M} \qquad (20c)$$

The Pricing Problem

The objective is changed as in (23c), which considers only the reduced costs of activity arcs.

$$\min z_{CCR}^{P,RS} = \sum_{t \in \mathcal{T}_m \setminus \mathcal{T}_m^{art}} \sum_{d \in \mathcal{D}_{w,t}} \sum_{e \in \mathcal{A}_{m,t,d}} C_e^r \cdot x_e^a \quad \text{(Reduced costs)} \qquad (23c)$$

s.t.

$$(11a), (12a), (24), (25), (26), (27), (29a)$$

7. Computational Results

In this section we begin with short introduction of a set of real-world instances from different German bus companies, followed by a description of the achieved solutions by using column generation, which are compared to existing methods. All experiments were conducted on an Intel Core i7-940 2.93 GHz processor with 32 GB RAM running Windows 7 Professional (64 bit).

7.1. Problem Data and Preliminary Tests

We use the same set of instances that are presented in Xie and Suhl (2013). These instances are available at the web page: `http://dsor.upb.de/crewrostering`. The number of duties in the instances varies between 1013 and 19486 duties, while the number of drivers in the instances varies between 48 and 629. The names of the instances include the number of drivers (NCCR) / groups of drivers (CCR), the planning days as well as the number of included shift types. In Table 1, the important properties of the instances are shown, including the number of activities, i.e, duties, standbys, days off, the average of alternatives, the number of activities arcs, the percent of fixed activity arcs in parentheses as well as the number of compatibility arcs. These are some of the factors causing difficulty for solving the instances. As we consider more activities (duties, standbys, days off) and drivers, more activity and compatibility arcs are generated in our network. If more alternatives for desired activities of drivers are provided, that also causes more activity and compatibility arcs. However, if we consider more fixed activity arcs, then this problem is easier to solve (such as the instances 96-70-8 vs. 89-70-8). The instances are sorted according to the gap to optimum after 24 hours for solving the integrated problem.

Besides those factors shown in Table 1, another important factor is how many quality rules are considered in those instances. Most of the instances in Table 1 consider all of the quality rule in

Section 2.1, but the instances 392-45-37, 393-45-37 and 397-40-37 only consider the objectives (1), (8), (9) as well as (10) and their corresponding soft constraints. It is one of the reasons why these instances are easier to be solved compared to for example 221-45-30.

Instance	Duties	Standbys	Days off	ϕAlternatives	Activity arcs (fixed)	Compatibility arcs
48-75-6	1,313	454	1,472	2.2	10,516 (12.7%)	34,584
52-73-6	1,288	664	1,487	2.2	11,367 (12.2%)	37,907
52-75-6	1,321	488	1,509	2.2	11,072 (15.5%)	36,113
9-238-11*	1,013	303	483	4.7	6,672 (12.4%)	30,457
393-45-37	5,420	2,619	8,030	1.5	45,035 (23.8%)	145,491
392-45-37	5,815	2,342	8,364	1.3	43,367 (26.2%)	134,974
397-40-37	4,917	2,717	7,438	1.4	38,277 (26.3%)	115,236
96-70-8	3,200	876	1,936	5	38,413 (6.3%)	287,490
87-70-8	3,273	796	1,708	4.5	38,944 (4.1%)	301,374
89-70-8	3,260	671	1,758	4.8	40,392 (4.8%)	326,505
221-45-30	4,214	1,646	3,222	6.7	64,576 (7.6%)	624,520
214-45-34	3,966	1,472	3,240	6.6	55,366 (8.7%)	472,958
211-45-34	4,003	1,447	3,029	7	55,376 (8.9%)	490,521
629-46-26	11,073	4,292	9,246	5	150,616 (11.8%)	1,212,486
606-70-26	18,739	4,977	13,612	5.3	255,657 (7.5%)	2,158,525
607-70-26	19,486	4,549	13,364	5.1	258,075 (7.3%)	2,205,303

Table 1: Characteristics of the test set instances. A star(*) indicates the instance for the CCR problem.

The instance 9-238-11 is used to solve the CCR problem. It includes nine groups of drivers (total 256 drivers) with different sizes, in which the maximum number of drivers in a group is 32, i.e., there are at most 224 planning days for a group. Seven artificial days are needed to expand the planning period, because there is a 14 day distance between two double-off activities. Therefore, we consider nine drivers in a planning period of at most 238 days. The instance size for the CCR problem is, in practice, usually small compared to the instance size for NCCR problems. The instance 9-238-11 is a large instance, which is available from the Germany bus companies.

Note that the weights of each objective are given by bus companies. The quality as well as the running time of the solution depends on the weights of the objectives. The details about the weights can be found in Xie and Suhl (2013) and different metaheuristics for solving the crew rostering problem are shown in Xie et al. (2013). According to Xie and Suhl (2013), the overview of the solutions for solving RS and IP by CPLEX solver are shown in Table 2. We take them as reference solutions for our new method.

instance	RS gap	RS time (sec.)	Int. gap	Int. time (sec.)
48-75-6	0%	6	0%	27
52-73-6	0%	6	0%	50
52-75-6	0%	5	0%	59
9-238-11	0%	7	0%	5,483
393-45-37	0%	17	1.68%	86,400
392-45-37	0%	17	9%	24,600*
397-40-37	0%	15	4.86%	86,400
96-70-8	0%	27,284	0%	48,100
87-70-8	0%	51,907	12.5%	86,400
89-70-8	39.8%	86,400	86.5%	86,400
221-45-30	42.9%	86,400	82.3%	86,400
214-45-34	5.8%	86,400	93%	86,400
211-45-34	13.7%	86,400	99%	86,400
629-46-26	10.9%	86,400	*	*
606-70-26	74.1%	86,400	*	*
607-70-26	97.7%	86,400	*	*

Table 2: Performance statistics of using CPLEX solver after 24 hours of computation for the rota scheduling (RS) and integrated planning (Int.). A star (*) indicates the available memory was exceeded.

7.2. Test Results of the Column Generation

The criteria for the termination of column generation are shown below.

- There is no new pricing solution with a negative objective value.
- The maximal running time is reached (in our case: 24 hours).

In our pricing problem, we generate feasible pricing solutions for drivers in parallel. According to our experiments, the Gurobi solver appears to deal with the parallelism better. Therefore, we test CPLEX and Gurobi solvers separately for solving column generation. After that, all inactive variables in column generation remain inactive and we solve the MIP problem as shown in Xie and Suhl (2013) with the CPLEX solver, but without the inactive variables in column generation. Therefore, the CPLEX solver solves the MIP problem with reduced variables. The CPLEX solver is chosen since it is better suitable to solve our MIP problem (see experiments shown in Xie and Suhl (2013)). Note that the default setting of the CPLEX and Gurobi solvers are considered in our tests, since parameter tuning for 16 instances will just overfit. In order to compare the different combination of solvers, the same initial solution is used for different tests of each instance, which is generated by our driver-by-driver heuristic.

Tables 3 and 4 show the solutions of our new approach compared to the reference solutions in Table 2. The instances which can be solved within a minute in Table 2 are ignored, i.e., the

instances 48-75-6 to 397-40-37 for the rota scheduling. For the same reason, the instances 48-75-6 to 52-75-6 for the integrated planning are not listed as well.

Instance	Solvers	Act.var.	Gap	Reduced	CPU(CG)	CPU(BB)	Reduced
96-70-8	C/C	94%	0%	0	1,355	15,286	-39%
	G/C	55.9%	0%	0	641	1,833	-65%
87-70-8	C/C	91.5%	0%	0	1,050	50,656	0%
	G/C	49.7%	0%	0	1,238	8,283	-81.7%
89-70-8	C/C	97%	39.5%	-0.3	928	85,455	0%
	G/C	52.7%	13.3%	-26.5	1,045	85,337	0%
221-45-30	C/C	92%	3.4%	-39.5	2,301	84,057	0%
	G/C	56.2%	0%	-42.9	5,759	80,598	-0.05%
214-45-34	C/C	94%	4.2%	-1.6	1,485	84,876	0%
	G/C	55.2%	0%	-5.8	950	85,250	0%
211-45-34	C/C	95.1%	9.37%	-4.33	1,744	84,607	0%
	G/C	52.9%	0%	-13.7	780	84,520	0%
629-46-26	C/C	92.2%	7.4%	-3.5	11182	74938	0%
	G/C	51.7%	0%	-10.9	3,711	4,736	-90%
606-70-26	C/C	90%	42.2%	-31.9	1,726	84,078	0%
	G/C	47.2%	39%	-35.1	371	85,399	0%
607-70-26	C/C	94.5%	93.7%	-4	13,527	72,388	0%
	G/C	55.3%	53.9%	-43.8	4,406	81,521	0%

Table 3: Performance statistics of using the column generation approach with the CPLEX (C) and Gurobi (G) solvers after 24 hours of computation of the rota scheduling.

Our column generation approach provides better solutions for both the rota scheduling and the integrated planning. For some instances up to 99% of the running time can be reduced (see last column: Reduced), while some achieve up to 76% smaller gap to optimum (see 5th column: Reduced). Some of them can get the both improvements, such as the instance 392-45-37. The instances (such as 89-70-8), which do not get any benefit of running time, stop due to the given running time (24 hours).

Additionally, our experiments show that better solutions can be obtained if we choose the Gurobi solver instead of the CPLEX solver as the solver for column generation. The reason for this is up to 40% less activated variables are selected at the end of column generation (see 2th column: Act.var.) with the Gurobi solver. Our experiments show that the CPLEX and Gurobi solvers choose different solutions after the first iteration in column generation. Therefore, different reduced costs are calculated in the master problem of each iteration.

Instance	Solvers	Act. var.	Gap	Reduced	CPU(CG)	CPU(BB)	Reduced
9-238-11	C/C	17.73%	0%	0	422	1930	-57%
	G/C	12.11%	0%	0	221	282	-90.8%
393-45-37	C/C	32.89%	0%	-1.68	3491	146	-95.8%
	G/C	28.06%	0%	-1.68	1197	60	-98.6%
392-45-37	C/C	27.96%	0%	-9	3457	59	-85.7%
	G/C	24.57%	0%	-9	1452	74	-93.8%
397-40-37	C/C	29.39%	0%	-4.86	2441	55	-97.1%
	G/C	26.40%	0%	-4.86	1165	137	-98.5%
96-70-8	C/C	37.28%	0%	0	11477	26689	-21.7%
	G/C	23.56%	0%	0	1937	1625	-92.6%
87-70-8	C/C	44.99%	0.37%	-12.13	16047	70118	0%
	G/C	24.4%	0%	-12.5	3679	3318	-91.9%
89-70-8	C/C	48.75%	84.4%	-2.1	20771	65365	0%
	G/C	22.35%	27.14%	-59.36	4334	81829	0%
221-45-30	C/C	39.33%	92%	9.7	27216	58611	0%
	G/C	21.39%	6.48%	-75.82	18108	67803	0%
214-45-34	C/C	36.59%	34.7%	-58.3	13366	72266	0%
	G/C	20.41%	30%	-63	746	85654	0%
211-45-34	C/C	39.95%	90%	-9	19506	66476	0%
	G/C	22.1%	86%	-13	7512	78888	0%
629-46-26	C/C	-%	-%	-	-	-	-%
606-70-26	C/C	-%	-%	-	-	-	-%
607-70-26	C/C	-%	-%	-	-	-	-%

Table 4: Performance statistics of using the column generation approach with the CPLEX (C) and Gurobi (G) solvers after 24 hours of computation of the integrated planning, a dash "-" indicates the solver ran out of memory.

	Unassigned duties(%)		Unassigned days(%)	
Instance	Seq.	Int.	Seq.	Int.
89-70-8	7.0	1.77	3.9	0
221-45-30	4.1	7.1e-4	2.9	0
214-45-34	4.2	0.53	2.9	0
211-45-34	4.9	7.5	3.5	0

Table 5: The comparison of sequential (seq.) and integrated (int.) solutions according to unassigned duties and unassigned days in percent.

According to comparisons between the solutions of sequential and integrated approaches in Xie and Suhl (2013), the main improvements of the integrated approach are more assigned duties and days. Table 5 shows the additional comparison for the instances, which obtain an optimal

sequential solution using the column generation approach (except the instance 89-70-8, which has a gap of 13.3%). The results of the integrated approach are not optimal, but they show about a 4% improvement of the percentage of unassigned duties and days over the result of the sequential approach. The exception is the instance 211-45-34, whose solution of the integrated approach is not good enough within 24 hours to compete with its sequential solution (gap of 86%).

The comparison of solutions obtained by different solution approaches for the rota scheduling problem is illustrated in Figure 8. The instances we consider here are from 96-70-8 to 607-70-26 in Table 1. The others can be solved with CPLEX within a minute, therefore, the improvement of them is not necessary. The solutions solved by the CPLEX solver are located at the point (0,0). The solutions of simulated annealing (SA) and Ant colony System (ACS) are obtained from Xie et al. (2013). The details of both approaches will not explain here. Although SA and ACS can get a solution within short time (for example: 30 minutes). However, the quality of the solutions for most of the solutions can not compete with CPLEX and our column generation approach. The metaheuristics are only suitable to be applied to solve very large instances, for instance 607-70-26, for which a good solution can be obtained by SA approach within 30 minutes.

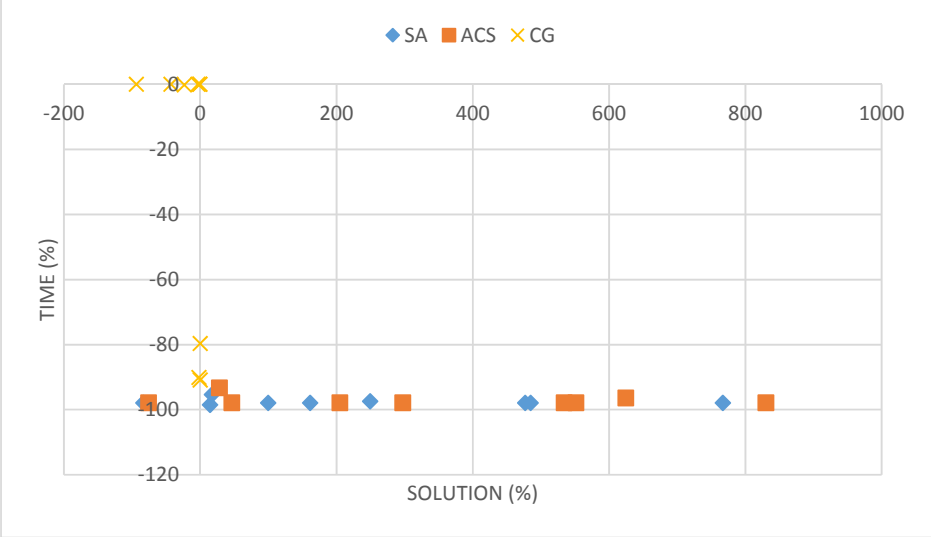

Figure 8 **Comparison solutions obtained by different methods (SA, ACS and CG) for the rota scheduling problem.**

8. Summary

This paper has described a heuristic method using column generation for solving the cyclic and non-cyclic crew rostering problem in public bus transit. For real-world instances, we obtain very good results compared to the results of commercial solvers, from the point of view of both solution time and gap to optimum of cost. For each instance either the solution time or the solution

quality is improved, or both. The solution times of some instances that can be solved within 24 hours, are improved an average by 90%. Up to 76% smaller gap to optimum is achieved by some of instances. Moreover, the results obtained for the unassigned duties and days clearly illustrate the benefits of the integrated method over the sequential one. Additionally, our column generation approach outperforms our previous work on metaheuristics for solving the subproblem rota scheduling problem.

Acknowledgments

The authors would like to thank the Moveo Software GmbH, especially Martin Finker, Matthias Kramp as well as Avar Schulze of that company for the helpful comments and fruitful discussions. Moreover, we would like to thank Dr. Kevin Tierney to improve the use of English in the manuscript.

References

Bianco, L., M. Bielli, A. Mingozzi, S. Ricciardelli, M. Spadoni. 1992. A heuristic procedure for the crew rostering problem. *European Journal of Operational Research* **58**(2) 272–283.

Cappanera, P., G. Gallo. 2004. A multicommodity flow approach to the crew rostering problem. *Operations Research* **52**(4) 583–596.

Caprara, A., M. Fischetti, P.L. Guida, P. Toth, D. Vigo. 1999. Solution of large-scale railway crew planning problems: The italian experience. *Computer-Aided Transit Scheduling*. Springer, 1–18.

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

Caprara, A., P. Toth, D. Vigo, M. Fischetti. 1998. Modeling and solving the crew rostering problem. *Operations Research* **46**(6) 820–830.

Catanas, F., J. Paixão. 1995. A new approach for the crew rostering problem. J. Daduna, I. Branco, J. Paixao, eds., *Computer-Aided Transit Scheduling*, *Lecture Notes in Economics and Mathematical Systems*, vol. 430. Springer, 267–277.

Day, P.R., D.M. Ryan. 1997. Flight attendant rostering for short-haul airline operations. *Operations Research* **45**(5) 649–661.

De Pont, G. 2006. Personalized crew rostering at netherlands railways. Master's thesis, University of Tilburg.

El Moudani, C.A.N., W.and Cosenza, M. de Coligny, F. Mora-Camino. 2001. A bi-criterion approach for the airlines crew rostering problem. *Evolutionary Multi-Criterion Optimization*. Springer, 486–500.

Emden-Weinert, T., H.G. Kotas, U Speer. 2000. DISSY-A driver scheduling system for public transport. Tech. rep., VSS GmbH and Bremer Straßenbahn AG, Bremen, Germany.

Ernst, A., M. Krishnamoorthy, D. Dowling. 1998. Train crew rostering using simulated annealing. *Proceedings of International Conference on Optimization Techniques and Applications*. Perth, 859–866.

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

Freling, R., R.M. Lentink, A.P.M. Wagelmans. 2004. 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.

Gamache, M., F. Soumis. 1998. A method for optimally solving the rostering problem. *OR in Airline Industry* 124–157.

Gamache, M., F. Soumis, G. Marquis, J. Desrosiers. 1999. A column generation approach for large scale aircrew rostering problems. *Operations Research* **47**(2) 247–263.

Hanne, T., R. Dornberger, L. Frey. 2009. Multiobjective and preference-based decision support for rail crew rostering. *IEEE Congress on Evolutionary Computation CEC'09*. 990–996.

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

Lee, C.K., C.H. Chen. 2003. Scheduling of train drivers for taiwan railway administration. *Journal of Eastern Asia Society of Transportation Studies* **5** 292–306.

Lučić, P., D. Teodorović. 1999. Simulated annealing for the multi-objective aircrew rostering problem. *Transportation Research Part A: Policy and Practice* **33**(1) 19–45.

Lučić, P., D. Teodorović. 2007. Metaheuristics approach to the aircrew rostering problem. *Annals of Operations Research* **155**(1) 311–338.

Maenhout, B., M. Vanhoucke. 2010. A hybrid scatter search heuristic for personalized crew rostering in the airline industry. *European Journal of Operational Research* **206**(1) 155–167.

Medard, C.P., N. Sawhney. 2007. Airline crew scheduling from planning to operations. *European Journal of Operational Research* **183**(3) 1013–1027.

Mesquita, M., M. Moz, A. Paias, J. Paixão, M. Pato, A. Respício. 2011. A new model for the integrated vehicle-crew-rostering problem and a computational study on rosters. *Journal of Scheduling* **14**(4) 319–334.

Monfroglio, A. 1996. Hybrid genetic algorithms for a rostering problem. *Software: Practice and Experience* **26**(7) 851–862.

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

Moz, M., A. Respício, M.V. Pato. 2009. Bi-objective evolutionary heuristics for bus driver rostering. *Public Transport* **1**(3) 189–210.

Pedrosa, D., M. Constantino. 2001. Days-off scheduling in public transport companies. S. Voß, J. Daduna, eds., *Computer-Aided Scheduling of Public Transport*, *Lecture Notes in Economics and Mathematical Systems*, vol. 505. Springer Berlin Heidelberg, 215–232.

Prakash, J., S.B. Sinha, S.S. Sahay. 1984. Bus transportation crews planning by goal programming. *Socio-Economic Planning Sciences* **18**(3) 207–210.

Respício, A., M. Moz, M.V. Pato. 2007. *A Memetic Algorithm for a Bi-objective Bus Driver Rostering Problem*. Centro de Investigação Operacional-Universidade de Lisboa.

Ryan, D. 1992. The solution of massive generalized set partitioning problems in aircrew rostering. *Journal of the Operational Research Society* **43**(5) 459–467.

Sellmann, M., K. Zervoudakis, P. Stamatopoulos, T. Fahle. 2000. Integrating direct CP search and CP-based column generation for the airline crew assignment problem. *Proceedings of the CP-AI-OR00*. 163–170.

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

Townsend, W. 1986. An application of the assignment model to bus crew rostering. *IMA Journal of Management Mathematics* **1**(1) 45–52.

Xie, L., N. Kliewer, L. Suhl. 2012a. Integrated driver rostering problem in public bus transit. *Procedia-Social and Behavioral Sciences* **54** 656–665.

Xie, L., M. Merschformann, L. Suhl. 2013. Metaheuristics approach for solving the crew rostering problem in public transit. Working paper, Nr. 1305, Univeristy of Paderborn.

Xie, L., M. Naumann, L. Suhl. 2012b. A stochastic model for rota scheduling in public bus transport. *Proceedings of 2nd Stochastic Modeling Techniques and Data Analysis International Conference*. 785–792.

Xie, L., L. Suhl. 2013. Cyclic and non-cyclic crew rostering problems in public bus transit. *OR Spectrum* Under Revision, no 2.

Yunes, T.H., A.V. Moura, C.C. De Souza. 2005. Hybrid column generation approaches for urban transit crew management problems. *Transportation Science* **39**(2) 273–288.