Column generation with a resource constrained shortest path algorithm applied to train crew scheduling

Master Thesis Econometrics and Management Science
Operations Research and Quantitative Logistics

Martine M.C. Verhave
358989

Erasmus University Rotterdam
Erasmus School of Economics
Rotterdam

Ab Ovo International
Business and software solutions
Capelle aan den IJssel

Faculty adviser:
Dr. T.A.B. Dollevoet
Second reader:
Dr. D. Huisman

Supervisors:
Drs. P. Koot
Drs. D. Veldhuis

15th December, 2015

Column generation with a resource constrained shortest path algorithm applied to train crew scheduling

Author:	Martine M.C. Verhave
Student number:	358989
E-mail:	358989mv@student.eur.nl
Content:	Master Thesis Econometrics and Operational Research - Operations Research and Quantitative Logistics
Study:	Master Econometrics and Operational Research
Faculty:	Erasmus School of Economics
Educational institute:	Erasmus University Rotterdam
Internship company:	Ab Ovo
Academic year:	2014 - 2015
Place:	Rotterdam
Date:	15th December, 2015
Faculty adviser:	Dr. T.A.B. Dollevoet
Second reader:	Dr. D. Huisman
External supervisors:	Drs. P. Koot and Drs. D. Veldhuis

ABSTRACT

This thesis deals with a large real world application of the crew scheduling problem. We formulate this problem as a set covering problem with additional constraints. The columns of the formulation correspond to shifts that are in accordance with the labour rules. The crew scheduling problem is proven to be NP-complete, and complete enumeration of all shifts is computationally intractable for most real-life applications. Column generation techniques are often used in literature to solve (integer) linear programs that are too large to consider all variables explicitly. In this thesis, we investigate the applicability of column generation to crew scheduling. The restricted master problem is the LP relaxation of the original program. Promising columns are generated inside the pricing problem. The pricing problem is formulated as a resource constrained shortest path algorithm. Performance improvements are obtained if we solve the pricing problem over different parts of the network. The independent pricing subproblems can be solved in parallel. The column generation technique provides good quality solutions within reasonable computation time for problem instances with up to 1500 tasks.

Key words: crew scheduling, set covering formulation, column generation, pricing problem, resource constrained shortest path

PREFACE

I am thankful for everyone who has supported me during my graduation.

My deepest gratitude is to my faculty adviser Twan Dollevoet. Thank you for your great and enthusiastic support, both during the programming phase and the writing process. You gave me a lot of time and expertise. I would like to thank my second reader Dennis Huisman as well. Thank you for giving your professional evaluation.

Of course, I would like to thank my external supervisors Dieter Veldhuis and Peter Koot. You gave me the opportunity to write my thesis at Ab Ovo International. It has been a pleasure working with you. Thanks, too, to everyone who made the staff weekend in Noordwijk an unforgettable experience.

I would like to express my gratitude to my family and friends for their unreserved love and support. I want to make a special mention for the support given by Damiën Horsten and Myrna van der Plas. Thank you for encouraging the use of correct grammar and commenting on this thesis.

At this point I would also like to direct a word of thanks to my fellow students. The past four years were a great experience, and I am grateful for the fantastic people I have met during those years. Now it is time to go our own way and show the world what we have learned from college.

Last but not least, I would like to thank you, reader, for your time and interest.

Martine Verhave
Rijnsburg, 2015

Contents

1 Introduction **1**

2 Planning process at the European rail-cargo company **3**

2.1 Overall planning process . . . 3

2.2 Crew scheduling at the European rail-cargo company . . . 4

3 Literature review **9**

3.1 Crew diagramming methods . . . 9

3.2 Airline crew scheduling . . . 10

3.3 Heuristics . . . 10

3.3.1 Run cutting heuristics . . . 10

3.3.2 Block cutting heuristics . . . 10

3.4 Metaheuristics . . . 11

3.4.1 Genetic algorithms . . . 11

3.4.2 Simulated annealing . . . 12

3.4.3 Tabu search . . . 12

3.5 Set covering approaches . . . 13

3.5.1 Lagrangian relaxation . . . 14

3.5.2 Column generation . . . 15

3.5.3 Combination of Lagrangian relaxation and column generation . . . 17

3.6 Conclusion of the literature review . . . 18

4 Problem formulation **19**

5 Methodology **25**

5.1 Crewblock construction . . . 25

5.1.1 Split large crewblocks based on cutting rules . . . 26

5.1.2 Block cutting and set partitioning . . . 27

5.1.3 Merge multiple workblocks into one crewblock to reduce the problem size . . . 29

5.2 Hand-over planning method . . . 32

5.3 Column generation . . . 33

5.3.1 General idea of column generation . . . 33

5.3.2 Restricted master problem . . . 34

5.3.3 Pricing problem . . . 34

5.3.3.1 Resource constrained shortest path algorithm 35
5.3.3.2 Dominance rules 37
5.3.3.3 Multiple pricing problems 41
5.3.3.4 Graph construction and resources 42

6 Data description **58**
6.1 Fixed parameter values 58
6.2 Problem instances for which the optimal solution is known 59
6.2.1 Problem instance one 60
6.2.2 Problem instance two 61
6.2.3 Problem instance three 62
6.2.4 Problem instance four 65
6.3 Problem instance of the medium-size European rail-cargo company 65
6.4 Problem instances derived from the real-life problem instance 68

7 Computational experiments **70**
7.1 Results for the problem instances with known optimal solution 70
7.1.1 Results for problem instance one 71
7.1.2 Results for problem instance two 73
7.1.3 Results for problem instances three and four 77
7.2 Sensitivity analysis 79
7.2.1 Crewblock construction 80
7.2.2 Effect of the dominance rule 84
7.2.3 Effect of a split per day of the week 87
7.2.4 Effect of the number of crewblocks 88
7.2.5 Effect of the data size and complexity of the break requirements 88
7.2.6 Effect of the number of employment groups 89

8 Conclusion and Further research **91**

References **97**

Appendices **100**

A Operational constraints for the real-life problem instance **100**

B Student's t-test **102**

1 Introduction

The market share of rail-cargo companies has been increasing consistently over the past few years (Woodburn (2012)). There is an increased level of competition within the rail-cargo sector. Due to governmental savings as a result of economic regression and privatization of transport companies, public financing of this sector is diminished and re-organization is forced. As a consequence, there is an increasing demand for computer-assisted approaches and decision support systems that can construct good quality train timetables, vehicle schedules and crew roster schemes automatically. In this paper, we focus on the crew scheduling process at a medium-size European rail-cargo company.

Each train operated by the company needs a driver. More specific, each *train activity* included in the vehicle schedule must be performed by an employee and is therefore called a *crew requirement*. Examples of train activities are driving operations, shunting operations, fueling operations, coupling operations and technical inspections of the train. A *shift* is a sequence of crew requirements and other activities for a single crew member to be executed in a day, including possible break moments (Qiao et al. (2010)). The crew diagramming problem (CDP) is to find a minimum cost set of *crew rosters* covering the crew requirements. A crew roster is a sequence of shifts carried out by one driver in a week, that is in accordance with the government regulations, collective labour agreements and union demands that are applicable to the driver.

In addition to the large set of operational requirements that must be fulfilled, it is important to pay attention to the requests of the company's employees. According to Abbink et al. (2005), the employees of the largest railway company of the Netherlands, NS Reizigers, were not satisfied with the new scheduling rules provided by the management in 2001. NS Reizigers introduced a set of rules called Circling-the-Church. Due to the introduction of this new set of rules, there was almost no variety in the activities performed by each employee, and some drivers and conductors got a large share of the unattractive part of the workload. This led to nationwide strikes in 2001 and an alternative set of rules, called Sharing-Sweet-and-Sour, was developed in cooperation with the employees of NS Reizigers. The introduction of this alternative set of rules satisfied the request of employees for more variety, and improved the trains' punctuality and the efficiency of NS Reizigers.

The CDP is commonly decomposed into the crew scheduling problem (CSP) and the crew rostering problem (CRP). This decomposition is described in detail by Hartog et al. (2009) and Caprara et al. (1997). The CSP is to find the minimum cost set of anonymous shifts that covers all crew requirements derived from the annual vehicle schedule and fulfills the operational constraints. Feasible crew rosters are constructed and assigned to the individual employees during the crew rostering phase. The total operation cost of a transport company can be reduced remarkably by using computer-aided approaches that construct crew roster schemes automatically instead of generating crew rosters manually, see Abbink et al. (2005) and Huisman et al. (2005).

In this paper, we focus on the CSP faced by a medium-size European rail-cargo company. Fischetti et al. (1989) have proven that the CSP is NP-complete, and the CSP can not be solved to optimality within a reasonable computation time for most problems that arise in practice. The explicit consideration of all feasible shifts is computationally intractable for most real-life CSPs. Column generation techniques and metaheuristics are examples of solution methods that work well in practice. We present a resource constrained shortest path (RCSP) algorithm that is used in the context of column generation to crew scheduling. We focus on the process of constructing crew schedules for train drivers, given the annual vehicle schedule. Other crew members fall outside the scope of this paper.

The remainder of this paper is organized as follows. The general planning process at railway companies is described in Chapter 2. The crew scheduling process at the medium-size European rail-cargo company is explained in detail in Chapter 2 as well. Chapter 3 provides an overview of existing literature on crew diagramming. The mathematical formulation of the problem is presented in Chapter 4. In Chapter 5, we discuss the outline of the methodology used to solve the CSP. The dataset provided by the medium-size European rail-cargo company is described in Chapter 6. Small problem instances, some derived from the original dataset, are used for sensitivity analysis. These datasets are also described in Chapter 6. The results of computational experiments are reported in Chapter 7. We end this paper with some concluding remarks and recommendations for future research.

2 Planning process at the European rail-cargo company

The planning process at railway companies is commonly decomposed into four planning levels. These planning levels are described in Section 2.1. Section 2.2 describes the crew scheduling process at the European rail-cargo company, including a detailed description of the operational constraints that must be fulfilled by the crew schedule.

2.1 Overall planning process

According to Huisman et al. (2005), the planning process at railway operators is commonly decomposed into four levels, namely the strategic, tactical, operational and dispatching level. The aim of this section is to describe these planning levels in more detail.

The strategic level takes place one or more years before the actual plan is taken into operation. At this level, the management team takes a critical look at the companies' current state. The management team focuses on achieving long-term objectives, and decisions are made based on the expected growth of the network and services. Rail fleet management software is used to determine the number and types of locomotives and wagons. The management team evaluates different scenarios of closing existing crew depots or opening new depot locations. Setup costs and union agreements are important factors that influence the decisions made by the management. The capacity of each crew depot is aligned with the expected future developments.

The annual *timetable* is built at the tactical level, one year to two months in advance. A timetable is a schedule of *train services* planned in a certain period, that satisfies the service level agreed on. A train service is the driving operation with cargo or passengers between two stations, specified by a departure and arrival time. Most railway companies design a timetable for one week that is representative for all weeks in the planning period. This week is called the *model week*. The timetable constructed for the model week is duplicated for the rest of the planning period. The cyclic timetable must be modified for some days, for example due to speed limitations as a result of track maintenance.

Two planning problems arise at the operational level, namely the vehicle scheduling problem (VSP) and the crew diagramming problem (CDP). The VSP is to assign rolling stock units, both locomotives and wagons, to cover all train services in the timetable. Several other train activities are planned, such as shunting operations and coupling activities. The CDP is to find the minimum cost set of crew rosters covering all crew requirements that are derived from the vehicle schedule. Crew rosters are constructed in accordance with the government regulations, collective labour agreements and union demands that are applicable to the crew.

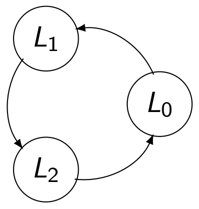
Real-time control of the overall planning is necessary to deal with disruptions, such as an accident at a certain part of the infrastructure which makes train traffic impossible on that line for a certain period of time. Decisions on how to deal with a disturbance need to be made in a short time period of just a few minutes, and heuristic methods are required (Huisman et al. (2005)). Crew requirements that remain unassigned due to disruption during daily operations or health issues of drivers are performed by available *reserve crews*. A reserve crew is an employee that stays at home or at a large station, ready to work if required. Reserve crew have minimum guaranteed hours paid, even if no crew requirement is performed. As a result, reserve crew is an expensive resource.

2.2 Crew scheduling at the European rail-cargo company

The CSP arises at the operational level. By then, the vehicle schedule for the model week is constructed. Each train activity must be assigned to one crew member. In this section, we introduce some terminology and give a description of the crew scheduling process at the European rail-cargo company. The operational constraints that must be fulfilled by the individual shifts, crew schedule and crew rosters are described.

Recall that the vehicle schedule covers all train services derived from the timetable. Rolling stock *deadheading* operations to reposition the rolling stock units between stations or to locate the units at a shunt yard are planned in the vehicle schedule as well (Caprara et al. (1997)). Some local train activities are scheduled, e.g., fueling operations, coupling operations and brake tests. The sequence of train activities planned on the same locomotive in the model week is called a *locline*. The model week starts at 12 AM on Sunday. Several loclines are scheduled in time behind each other, so that a circulation of one or more weeks is formed. This circulation is called a *locline cycle*. Each locline is included in exactly one locline cycle. The same locomotive is used to operate the train activities planned on successive loclines in the cycle in consecutive weeks. Figure 1 shows a schematic representation of a locline cycle with three loclines. The locomotive that is assigned to locline L_i in the first week is assigned to locline L_j in the p-th week, with $i \in \{0,1,2\}$, $p \in \mathbb{Z}_{\geq 0}$ and $j = (i + p - 1)(\text{mod } 3)$.

Figure 1: Example: Locline cycle with three loclines



The set of all train activities planned in the model week is given by T and the set of locline cycles is given by LC. Cycle $l \in LC$ is represented by the sequence of train activities planned in time behind each other on the loclines in l, i.e. $l = \{t_{1,l}, ..., t_{n_l,l}\}$ where $t_{i,l} \in T$ for all positions $i \in \{1, ..., n_l\}$ and n_l the number of train activities in locline cycle l. The end location of train activity $t_{i,l}$ must be equal to the start location of activity $t_{(i)(mod\ n_l)+1,l}$, for all positions $i \in \{1, ..., n_l\}$. Furthermore, these train activities can not overlap in time. A locline cycle is only workable if it is in accordance with the conditions mentioned above.

The crew requirements that can be executed by a driver are defined based on the *traction knowledge*, *route knowledge* and *skills*. The driving license of a crew member determines the locomotive types that the employee can drive. This information is called traction knowledge, see Laplagne (2008). Similarly, crew members are trained to operate on certain parts of the network and they can only drive trains on routes that they are familiar with. The employee's skills determine the types of crew requirements that the driver can perform. There are two driver *work functions*, namely local drivers and global drivers. Local drivers have knowledge of a small part of the network and can only fulfill certain crew requirements. The traction knowledge, route knowledge and skills of global drivers are more expanded. The *work regime* of an employee specifies the collective labour agreements and union demands that are applicable to the driver.

Each driver is connected to exactly one crew depot, which is the employee's *home base*. The other crew depots are *away locations* of this employee (Vaidyanathan et al. (2007)). An *employment group* is a group of drivers who have the same home base location, work regime and work function. For simplicity, we assume that all drivers of an employment group have the same route knowledge, traction knowledge and skills. In practice, this will not be the case and the planning must be adjusted manually to correct for this.

Shifts with a duration of at least 3 hours between 10 PM and 6 AM are *night shifts*. An *international shift* starts from or ends up at a crew depot that is located in another country than the home base location of the crew member who operates the shift, or it covers at least one international crew requirement. These two characteristics define the *shift type*. Night shifts are valued less attractive than regular day shifts, and drivers claim an additional payment to cover these unattractive shifts. Further, drivers claim an additional payment if they must work on Monday before 4 AM or on Friday after 7 PM.

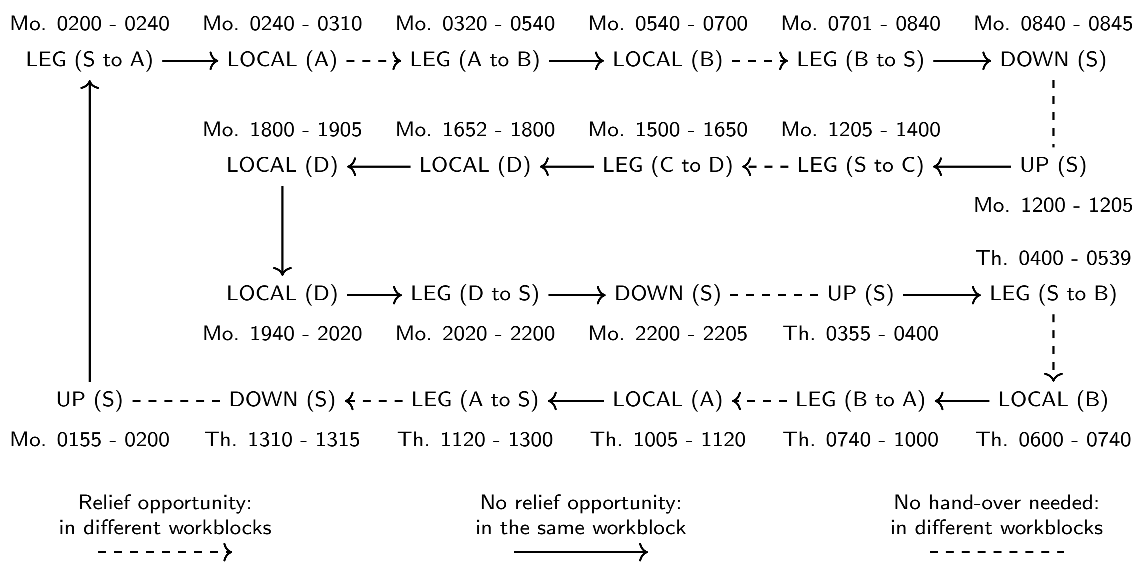
The rolling stock units may not be left unattended, unless the driver has performed a shut-down activity to secure the unit whilst unmanned. It is likely that this option makes the schedule more efficient, since the driver can be used to cover other crew requirements. The equipment and engines must be checked if the locomotive is restarted. The shut-down and start-up activities take a non-negligible amount of time to execute. These activities are planned in the vehicle schedule. A locomotive that can be left unattended is called an *immobilized* locomotive. A *mobilized* locomotive is not allowed to be unmanned.

Relief locations are the subset of all locations where a crew change on a mobilized locomotive can be planned (Banihashemi and Haghani (2001)). The *hand-over* of responsibility between the drivers involved in a crew change on a mobilized locomotive ensures that the locomotive is not left unattended. A *relief opportunity* exists between two crew requirements planned in sequence on the some locomotive if the end location of the first activity is a relief location, and the idle time between the crew requirements is larger than or equal to the hand-over duration. The duration of a hand-over does not depend on the relief location, and is equal to 1 minute. A window of relief opportunities exists if the time gap before the train departs from a relief location is larger than 1 minute. More information about relief opportunities can be found in Laplagne (2008).

The sequence of crew requirements between two consecutive relief opportunities in a locline cycle is called a *workblock*. These crew requirements need to be performed by the same driver. This driver must have the route knowledge, traction knowledge and skills to perform all crew requirements in the workblock. A schematic representation of a locline is shown in Figure 2. No other locline is planned in time behind this locline. Local train activities, such as coupling operations and brake tests, are labeled by the term LOCAL. The term LEG is used to refer to the driving operations with cargo and the vehicle deadheading operations. Shut-down (DOWN) and start-up (UP) activities are planned if the rolling stock unit is left unattended. No driver can transfer from one rolling stock unit to another one at location D. The start time and end time of each train activity are depicted. For example, the shut-down activity specified by 'Th. 1310 - 1315' is scheduled on Thursday from 13:10 till 13:15.

A disruption during daily operations can have consequences both on the vehicle schedule and crew schedule. Suppose that a train is delayed. The driver who takes over the locomotive must wait at the relief location for the train to arrive, and it is reasonable that both drivers involved encounter problems during the rest of their working day. To reduce the consequence of disturbances on the crew schedule, the number of crew changes on a mobilized locomotive is limited. The planner can combine multiple workblocks planned in time behind each other on the same locline cycle into one *crewblock*. The crew requirements in a crewblock are performed by a driver who has the route knowledge, traction knowledge and skills to perform each activity in the elementary sequence. The advantage of combining multiple workblocks into one crewblock is the reduced size of the problem, which will lead to shorter computation times. A drawback is that the solution quality might depend on the number of workblocks that are combined into one crewblock. In particular, one might be able to reduce the cost of the solution by decreasing the number of workblocks that are combined into one crewblock.

Figure 2: Example: Locline cycle with one locline



Crew deadheading refers to the repositioning of crew between two relief locations. A taxicab or car is used to transfer crew between relief locations that are far apart, while a bicycle and by foot are suitable to travel between neighboring locations. Shifts can also contain *passenger activities*, meaning that a driver is traveling as a passenger on a train. If more than one driver is allocated to a locomotive at the same time, we assume that one qualified driver performs the actual train activities and that the other crew members are passengers on the train. Passenger activities are planned on individual train activities. In case of the European rail-cargo company, each driving operation can be used for crew deadheading. See Vaidyanathan et al. (2007) for more information about crew deadheading.

Each individual shift observes a set of working rules, concerning shift hours, driving time regulations and required breaks. The following issues need to be addressed:

- The duration of the shift, also called the *spread time* of the shift, must be within some predefined time interval. This interval depends on the work regime and work function of the driver who operates the shift. Further, the time interval may be different for national and international shifts.
- The break requirements depend on the work regime and work function of the driver who operates the shift, the type and spread time of the shift. The total break time required, the

minimum duration per meal break and the time window in which the first break must take place after the start of the shift are given. It is possible to have a break on the locomotive or during crew repositioning.

- The total duration of all driving operations in one shift may not exceed the maximum driving time, which depends on the work regime and work function of the employment group to which the shift is allocated, and the shift type. Passenger activities are not considered as driving time.
- All shifts operated by a local driver start and end at the home base location of the driver. Global drivers may rest in a hotel at an away location for a maximum of one consecutive night, i.e. the number of *overnight* stays is limited to one for global drivers.
- A log in activity of 10 minutes is required at the start of a shift. Furthermore, each shift ends with a log out activity of 5 minutes. Both *crew activities* are planned at a crew depot.

Let D be the set of employment groups. The crew roster scheme is subject to a number of comprehensive constraints, concerning the percentage of overnight stays, regulating crew capacities, etc. These rules involve multiple shifts at the same time. The following set of constraints must be fulfilled.

- Consider employment group $d \in D$. Let $\underline{V}_d$ and $\bar{V}_d$ be the minimum and maximum average weekly workload per driver of employment group d, respectively. The average weekly workload per driver of the employment group, $\langle V_d \rangle$, is equal to the total spread time of the shifts allocated to group d divided by the number of crew necessary to operate these shifts. It must hold that $\bar{V}_d \geq \langle V_d \rangle$. Further, the management team prefers that $\langle V_d \rangle \geq \underline{V}_d$. This is taken into account in the objective function, i.e., a penalty cost is incurred if $\langle V_d \rangle < \underline{V}_d$.
- The minimum average number of rest days per driver per week is 2. This is computed per employment group.
- Let f_d be the maximum fraction of shifts allocated to employment group $d \in D$ with an overnight stay at the end. It must hold that $f_d = 0$ if employment group d consists of local drivers. Global drivers must return to their home base location at least every two shifts, and it must hold that $0 \leq f_d \leq 0.5$ if global drivers are part of employment group d.

Some rostering aspects can only be taken into account during the crew rostering phase. For example, the connection time between two adjacent shifts performed by the same employee should satisfy the corresponding standards. The minimum and maximum required rest time depend on whether the crew member spends this time at home or at a hotel away from his home base. Furthermore, the weekly workload of a driver that is part of employment group $d \in D$ may not exceed the maximum weekly workload, which is larger than or equal to $\bar{V}_d$.

3 Literature review

In this chapter, existing literature on crew scheduling and crew rostering is reviewed. First, some papers are presented that review methods and algorithms that are used to solve the CDP. The differences between the airline CSP and the railway CSP are described in Section 3.2. Heuristics, metaheuristics and set covering approaches that are used in literature to solve CSPs are described in Section 3.3, Section 3.4 and Section 3.5, respectively. Finally, we position the scope of our analysis within the existing literature and give a conclusion on how the reviewed literature can be used.

3.1 Crew diagramming methods

The CDP has been studied extensively in the past, especially for airline and mass-transit crew scheduling applications. The need for efficient computerized crew scheduling algorithms to solve the CDP faced by railway companies has increased over the past three decades, as the complexity of the planning puzzle grew and manual planning became intractable.

The CDP is a challenging planning problem faced by many companies, such as transport companies, call centers and hospitals. Each company has unique characteristics, resulting in a need for specific mathematical models and algorithms to solve the staff rostering problem in different areas of application. An overview of almost 700 studies in the area of crew diagramming that have been published since 1950 is given by Ernst et al. (2004a). Most of these papers focus mainly on the crew scheduling and crew rostering phase, but some also cover workforce planning and other related areas. The papers are classified according to the type of problem addressed, the application areas covered and the methods used. Ernst et al. (2004b) review scheduling and rostering methods and comment on the applicability of the techniques for solving the CDP in different application areas. The authors mention that literature is skewed towards mathematical programming and metaheuristic approaches.

Sodhi and Norris (2004) present a flexible and fast modeling approach in which the CDP is divided into stages that can each be solved with a standard mixed integer linear program (MILP) solver or manual approaches. Three types of shifts exist, i.e. early, late and night shifts. The first stage of this approach is decomposed into three phases and the aim is to construct the rest-day pattern including shifts. A single graph is created across all depots and a MILP model is solved to produce the patterns that contain an 'optimal' rest-day pattern for a specific depot. A depth-first search heuristic is used to find the rest-day pattern for a specific depot using the output of the MILP model. In the second stage, concrete shift are assigned to the created rest-day pattern. This problem is modeled as an assignment problem with side-constraints. Sodhi and Norris (2004) describe an application at the London Underground. They use data from different depots, with crew sizes ranging from 30 to 150 drivers. Solving the MILP model in the first stage takes a few minutes at most and it is solved within 2% from optimality. The computation time for the second stage is only a few seconds.

3.2 Airline crew scheduling

Several review papers are available that address crew planning methods and algorithms with applicability in the airline sector, including Souai and Teghem (2009) and Sellmann et al. (2002). In general, the railway CSP is more complex than the airline CSP, as more activities are planned in the train schedule than in the aircraft schedule. The number of feasible shifts is larger for railway companies. Nevertheless, some methods used for airline applications can solve the rail CSP within a reasonable time limit.

3.3 Heuristics

Heuristic methods based on rules used by manual planners were extensively used in the past to solve the CSP. It was not possible to solve reasonable size problem instances with the technology available then. Nowadays, heuristics are used to reduce the problem size so that mathematical programming can be used to find a solution within a reasonable time limit. Laplagne (2008) claims that heuristics are needed in at least one stage to solve large-size CSP instances with mathematical approaches presented in literature.

3.3.1 Run cutting heuristics

Run cutting techniques are based on the idea used by manual planners to cut locline cycles into crewblocks using cutting rules. Thereafter, two or three of these crewblocks are combined into one shift based on their combination quality. The quality of a crewblock combination is evaluated in terms of the number of driving operations covered and in terms of the time left for subsequent crew requirements. Each shift must obey the relevant working rules. All crewblocks are included in at least one shift and the cost of the solution is calculated based on shift characteristics. The solution quality can be improved by using other criteria to cut the locline cycles into crewblocks and by exchanging crewblocks between the shifts. The run cutting approach is often used to generate an initial feasible schedule, see Cavique et al. (1999).

3.3.2 Block cutting heuristics

An algorithm for the CSP with bin packing features is presented by Qiao et al. (2010). The crew scheduling process is decomposed into two parts, namely block cutting and piece matching. The block cutting process cuts the locline cycles only at relief locations, and the minimum and maximum length of a crewblock are set to reduce the problem size. There are different cutting patterns, as the locline cycles can be cut at several points. All feasible blocks are generated given the set of cutting patterns and a large candidate list of crewblocks is formed which covers each crew requirement at least once. The piece matching process is formulated as a bin packing problem and the objective is to minimize the total costs of the bins, i.e. the shifts, such that all crew requirements are covered and the operational requirements are fulfilled. A modified best-fit-decreasing heuristic is used to find a solution for the piece matching process.

Ball and Benoit-Thompson (1988) provide an approach that iterates between the solution of a shortest path problem and the solution of a matching problem to solve the CSP. The block partitioning problem is formulated as a shortest path problem and the non-linear objective function of this problem is updated based on information obtained from the solution of the matching problem at each iteration. The solution of the matching problem is a set of shifts that covers all crew requirements at least once.

3.4 Metaheuristics

Research in metaheuristics and their application has become very popular over the last twenty years, see Laplagne (2008). In this paper, the following definition of a metaheuristic is used.

"A metaheuristic is formally defined as an iterative generation process which guides a subordinate heuristic by combining intelligently different concepts for exploring and exploiting the search space, learning strategies are used to structure information in order to find efficiently near-optimal solutions." (Blum and Roli (2003))

3.4.1 Genetic algorithms

Genetic algorithms rest on the idea of natural selection, stated by the English naturalist and geologist Charles Darwin (1809 - 1882). First, a large candidate list of feasible shifts is constructed. The population of candidate solutions will evolve to a population with a better solution quality, by using mutation and crossover techniques to improve the candidate solutions. Each population member is represented by a chromosome, which contains one bit per shift. The bit is one if the shift is selected in this candidate solution. The weakest candidate solutions in the population are replaced by new offspring in each iteration. Crossover, mutations and repair heuristic are applied to the newly added members, such that enough variation is present in the solution space. A detailed description of genetic algorithms can be found in the paper written by Beasley and Chu (1996).

Souai and Teghem (2009) describe an approach based on a hybrid genetic algorithm that solves the two subproblems of the CDP simultaneously. First, a subset of feasible shifts is generated for each day in the planning horizon by solving a set partitioning problem (SPP) or by using a graph theoretic heuristic. Second, the selected shifts are assigned to the available crew members in order to build a solution in which all operational constraints are fulfilled. It might occur that some shifts can not be assigned to a driver. Multi-point crossover operations and mutations are used to change one or more solutions in the population. A legality repair heuristic and feasibility repair heuristic are described that can be used to make the solution legal and feasible, respectively. Three real-world problem instances for airline crew-pairing and rostering were solved within 2 hours of computation time.

3.4.2 Simulated annealing

Emden-Weinert and Proksch (1999) present a simulated annealing algorithm for the airline crew pairing problem. A run cutting formulation is used to construct a set of pairings. Computational results are given for real-word test problems with up to 4600 flights per month. Combining simulated annealing with a problem specific local improvement heuristic can reduce the run time, while improving the solution quality.

3.4.3 Tabu search

Tabu search algorithms are methods employing local search methods used for mathematical optimization. Neighboring solutions are explored and worse solutions can be accepted if it is not possible to improve the solution by a move. In this way, the algorithm does not get stuck in a local optimum. Previous visited solutions are stored on the tabu list. It is not allowed to select any solution appearing in the tabu list as the next solution. The tabu list is dynamically updated during the search. The best found solution is returned when the algorithm terminates. For a technical study on tabu search algorithms, see Glover (1989) and Glover (1990).

Cavique et al. (1999) propose a tabu search algorithm for the CSP that is based on a run cutting approach. The run cutting heuristic is used to construct an initial schedule and the tabu search heuristic is used during the improvement phase, where the goal is to reduce the number of shifts selected in the solution. After each iteration, expensive shifts are quashed and the run cutting process is used to construct a set of shifts for the subproblem. A tabu list is used to avoid the creation of shifts that were previously removed from the solution. Cavique et al. (1999) also describe a sub-graph ejection chain method based on the formulation of the CSP as the maximum cardinality matching problem on a non-bipartite graph. This method starts from an initial solution and attempts to improve the current solution iteratively. As the number of possible graphs is large, a tabu search framework using compound neighborhoods is used to solve the problem.

3.5 Set covering approaches

The CSP is often defined as a set covering problem (SCP) or a set partitioning problem (SPP), see for example Jütte and Thonemann (2012). The columns of this formulation are shifts that are in accordance with the government regulations and labour rules. The explicit construction of all feasible shifts is computationally intractable for most real-world problem instances, and linear programming or branch-and-price algorithms are needed, as mentioned by Qiao et al. (2010). The problem size is reduced by merging several workblocks into one crewblock. A large set of candidate shifts is constructed and added to the formulation. The problem is solved with the current set of shifts.

Let S be the set of candidate shifts added to the formulation. The set of crewblocks that must be covered is given by C. The binary decision variable X_s indicates if shift $s \in S$ is selected in the final solution (1) or not (0). The cost to operate shift $s \in S$ is given by c_s. Binary parameter $a^1_{c,s}$ indicates if crewblock $c \in C$ is included in shift $s \in S$ (1) or not (0). We assume that every crewblock must be assigned to at least one driver, and that the minimum-cost set of shifts can only be found if each crewblock is included in at least one shift in S. The set covering (set partitioning) solution is the minimum-cost subset $Z \subseteq S$ of shifts, such that each crewblock is included in at least (exactly) one shift in Z. There are a lot of extra constraints compared to a standard SCP and the following SCP with side-constraints is used.

$$min \qquad \sum_{s\in S} c_s X_s \tag{3.1}$$

$$s.t$$

$$\sum_{s\in S} a^1_{c,s} X_s \geq 1 \qquad \forall c \in C \tag{3.2}$$

$$\sum_{s\in S} g_{p,s} X_s \leq d_p \qquad \forall p \in F = \{1, ..., f\} \tag{3.3}$$

$$X_s \in \{0,1\} \qquad \forall s \in S \tag{3.4}$$

The objective is to minimize the total cost associated with the solution (3.1). Constraints (3.2) indicate that every crewblock is covered by at least one selected shift. If a crewblock is included in more than one selected shift, we assume that one crew member performs the actual crew requirements and that the other crew members are passengers on the locomotive. There is equality in constraints (3.2) in the set partitioning formulation, so that each crewblock is covered exactly once. Constraints (3.3) are additional constraints based on government regulations and labour rules. For instance, one can limit the number of shifts selected in the solution. The shifts in the candidate solution obey the naturally occurring constraints, and rules for individual shifts are not taken into account in (3.1) - (3.4).

Willers et al. (1995) present a model for solving the bus driver scheduling problem. This model is a combination of set partitioning and set covering constraints with additional constraints, as driving operations at the start and end of the day are never overcovered. First, the number of shifts selected is minimized by using a primal steepest edge algorithm starting with an initial set of shifts. Secondly, a constraint is added to the formulation which ensures that the number of shifts can not increase and a primal steepest edge algorithm is used to minimize the total cost of the shifts selected. As a consequence, minimizing the number of shifts has absolute priority. A dual approach is presented, which is faster than the primal strategy for 18 out of 20 real-world problem instances.

The paper written by Smith and Wren (1988) formulates the bus driver scheduling problem as a SCP. The goal is to minimize the total number of shifts, such that all driving operations are covered at minimum cost and all relevant side-constraints are fulfilled. A linear relaxation of the set covering formulation is solved and constructive heuristics reduce the problem size. A branch-and-bound method is used to find an integer optimal solution of the CSP that meets all side-constraints.

Recall that complete enumeration of all feasible shifts is computationally intractable for most real size problem instances. The Lagrangian relaxation method and the column generation technique can be used to reduce the computation time. The Lagrangian relaxation approach is described in Subsection 3.5.1. The idea of column generation and papers that use this technique to solve the CSP are presented in Subsection 3.5.2. The studies presented in Subsection 3.5.3 describe methods that use a combination of these two techniques.

3.5.1 Lagrangian relaxation

The idea of the Lagrangian relaxation approach is to relax some of the difficult constraints in the set covering or set partitioning formulation, and to penalize their violations in the objective function. The Lagrangian relaxation method is used to find a lower bound on the optimal solution of the SCP or the SPP. More information about Lagrangian relaxation and optimization can be found in Fisher (2004) and Qiao et al. (2010).

Caprara et al. (1999) present a Lagrangian-based heuristic to solve the SCP. A 3-phase heuristic is used to improve the solution of the SCP in consecutive iterations. In the sub-gradient phase, a near-optimal Lagrangian multiplier vector is found by using a sub-gradient algorithm. A sequence of near-optimal Lagrangian multiplier vectors is constructed in the heuristic phase, given the solution of the first phase. A heuristic solution to the SCP is computed for each of the Lagrangian vectors, and the current best solution found is updated each time a better solution is found. Columns that have a high probability to be selected in the optimal solution are set equal to 1 in the column fixing phase. The 3-phase heuristic is repeated until it is not possible to find a better solution for the SCP, i.e. if all crew requirements are covered by the set of fixed columns or if the sum of the costs of the fixed columns plus a lower bound on the cost of the remaining problem is at least the value of the best found solution value of the SCP. New columns are added after solving the pricing problem. After finishing the 3-phase heuristic, a refining procedure is performed that fixes some columns and re-optimizes the resulting subproblem by using the 3-phase heuristic. The process is repeated until a sufficient precision is obtained or until a given time-limit is exceeded. The algorithm provided by Caprara et al. (1999) finds near-optimal solution for the CSP within a reasonable time limit for large instances coming from real-world railway applications.

3.5.2 Column generation

The column generation technique is introduced by Dantzig and Wolfe (1960). This technique is often used to solve (integer) linear programs that are too large to consider all variables explicitly. The problem is split into two subproblems, namely the restricted master problem (RMP) and the pricing problem. The RMP is the (linear programming (LP) relaxation of the integer) linear program in which only a subset of variables is considered. Variables that have the potential to improve the objective function are generated in the pricing problem, via a constraint-shortest-path, k-shortest path or constraint-programming approach (see Guo et al. (2006)). These variables are added to the formulation. The optimal solution of the RMP is found if no variable with negative reduced cost is identified in the pricing problem. In our case, the master problem is the LP relaxation of the SCP or SPP. The solution of the RMP gives a lower bound on the solution value of the original problem. The combination of column generation and branch-and-bound techniques is called branch-and-price. For a detailed description of branch-and-price in the context of crew scheduling, see Huisman (2007).

Jütte et al. (2011) describe a column generation based optimization software package that is able to generate high quality solutions for the CSP in a few days for problem instances with up to 30.000 crew requirements. Initially, shifts are generated that have a negative reduced cost. The master problem, which is the LP relaxation of a mixed-integer optimization problem, is solved with the candidate set of shifts. Additional shifts are identified by the pricing problem until an integer solution is found for the master problem.

Although high-quality solutions can be obtained within reasonable run times, Jütte and Thonemann (2012) mention that the run times needed by the algorithm of Jütte et al. (2011) are too long to allow for short-term planning. Especially in the rail-cargo business, crew schedules must be generated quickly as a large portion of the trains is scheduled few days before operation. Jütte and Thonemann (2012) propose a divide-and-price approach which decomposes the overall problem into overlapping subproblems. Each crewblock is assigned to one primary subregion and possibly multiple secondary subregions. The number of subregions must be chosen such that there is a balance between the solution quality and the solution time. The subproblems are solved in parallel and the subregions are updated after several iterations. The authors use a column generation approach that solves a resource constrained shortest path (RCSP) problem, followed by pricing and assignment updates. Crewblocks are finally assigned to the subregion where they can be covered at lowest cost. The run times are reduced significantly compared to the run times attained with the approach described by Jütte et al. (2011), while the quality loss is low.

Jütte and Thonemann (2012) show that decomposing the problem instance in multiple subproblems affects the run time significantly. Based on this observation, Jütte and Thonemann (2015) present considerations on how to decompose the problem instance into smaller subproblems. A graph partitioning based decomposition algorithm is deducted with four different decomposition variations. A column generation approach combined with a variable fixing technique is used to solve the subproblems simultaneously. The decomposition variation based on productivity showed the best performance for real-life experiments.

Banihashemi and Haghani (2001) present a relaxed model for the mass-transit crew scheduling (MTCS) problem based on the multi-commodity formulation of the multi-depot vehicle scheduling (MDVS) problem and a constraint generation approach. The MTCS problem is formulated as a SCP. Different types of shifts, e.g., long and short shifts, are considered as different types of commodities and the formulation is task-based. Hard and soft constraints are added to the relaxed MTCS problem, which respectively restrict the construction of specific shifts or add a penalty to the objective function if specific shifts are constructed. The relaxed MTCS problem is solved to optimality and a list of feasible shifts is built based on all constraints. The procedure to solve the relaxed MTCS to optimality is a row-column-generation approach. Unfortunately, this procedure can only be applied to very small problems with a maximum of 30 tasks.

Freling et al. (2001) propose a heuristic branch-and-price algorithm that can be used to solve large scale CSPs. The first step of this algorithm is to generate an initial set of feasible shifts. The LP relaxation of a set covering formulation is solved given the candidate shifts, and additional shifts are obtained by using dual information from the solution. This process is repeated until the LP relaxation is solved to optimality. The procedure to solve the IP problem uses the same column generation algorithm to solve the LP relaxation in every other node of the branch-and-bound tree. The authors propose three algorithms that are used to construct feasible shifts. The branch-and-price algorithm proposed by Freling et al. (2001) can also be applied, after small modifications, for solving CSPs in areas other than the train sector. Large scale CSPs can be solved within a reasonable time limit and it is easy to implement labour rules and governmental regulations. Techniques to speed up the algorithm are implemented, e.g., an acceleration technique that reduces the network size by removing invalid arcs.

Desrochers and Soumis (1989) also describe a heuristic branch-and-price algorithm. New feasible shifts are proposed to improve the current solution of the SCP by solving a shortest path problem with resource constraints defined on an acyclic network. This subproblem is solved using a dynamic programming algorithm. The planner must specify which columns with negative marginal cost must be added to the set of candidate shifts. The performance of the algorithm is evaluated for two relatively small real-life problem instances.

Savelsbergh (1997) presents a branch-and-price algorithm of the generalized assignment problem (GAP) that examines the maximum profit assignment of n tasks to m crew members, such that each task is assigned to one crew member and all shifts are feasible. The authors mention that solving an LP relaxation of an IP problem with column generation does not necessarily give a feasible solution for the original IP formulation, until the decision variables of the relaxed problem are integer. The pricing problem must be such that columns that are infeasible due to the branching constraints will not be generated. The authors consider various column generation schemes, and two branching strategies are explored. A primal heuristic is used at each node of the branch-and-bound tree, which uses the current solution of the LP problem to measure the desirability of assignments. Four experiments are used to evaluate the performance of the branch-and-price algorithm for small problem instances with a maximum of 20 crew members and 40 tasks.

3.5.3 Combination of Lagrangian relaxation and column generation

Abbink et al. (2008) provide a mathematical formulation of the CSP as an SCP with side-constraints. Crew requirements and shifts are for two different planning days. The automatic crew scheduling algorithm used by the largest railway company of the Netherlands solves this CSP by using column generation combined with the Lagrangian-based heuristic provided by Caprara et al. (1999). Different partitioning methods are presented. The global constraints are to be validated on a weekly basis. In some cases the solution can be improved, for instance by re-scheduling the solution for one crew base. Weekday, geographical and line based partitioning methods are presented. Partitioning based on column information is the fourth partitioning method presented by Abbink et al. (2008). The partitioning methods give promising results for a set of experiments.

Abbink et al. (2011) present an algorithm that combines Lagrangian heuristics, column generation and fixing techniques. Promising shifts are constructed by solving a shortest path algorithm over different parts of the network. They split the cyclic network into seven subnetworks, each associated to a particular weekday. The subnetwork of a weekday overlaps with the subnetwork of the next weekday, such that all feasible shifts can be constructed. Computational benefits can be obtained by running the algorithm in parallel. The authors extended the algorithm to solve instances for the complete week. Promising results were found.

3.6 Conclusion of the literature review

Heuristic methods are often used to reduce the problem size so that (integer) linear programs can be used to find a solution to crew scheduling within reasonable computation time. We use the idea of manual planner to cut the locline cycles into crewblocks based on cutting rules. The advantage of this approach is the reduced size of the problem, and the CSP is easier and faster to solve. Usually, there is a trade-off between the problem size and the solution quality, since the number of feasible shifts reduces when multiple workblocks are combined into one crewblock.

Crew scheduling has been studied in different fields with many different approaches. Most problem instances that are used to test the methods and algorithms described in literature are significantly smaller than the crew scheduling problem faced by the medium size European rail-cargo company. More than 15,000 crew requirements must be covered each week. Most techniques presented in literature can not be used to solve large problem instances, even though acceleration techniques are developed that reduce the size of the problem.

Several studies have shown that Lagrangian relaxation methods and column generation techniques reduce the run time significantly. We present a column generation approach that is used to solve the train driver scheduling problem. The master problem is formulated as a set covering problem with additional constraints. The pricing problem, formulated as a resource constrained shortest path problem, is used to propose new feasible shifts that have the potential to improve the objective function of the RMP. The objective function of the pricing problem is the reduced cost of the solution variable with respect to the current dual values of the constraints in the RMP. This idea was also used by Desrochers and Soumis (1989).

Solving the pricing problem can be very time consuming. Significant performance improvements can be obtained by splitting the pricing problem into multiple, smaller pricing subproblems. Instead of solving the pricing problem for the entire week, Abbink et al. (2011) split the pricing problem on weekday. Each subgraph is associated with a particular day of the week. The independent pricing subproblems can be solved in parallel, and the number of threads depends on the number of CPU cores available. We use the idea to split the pricing problem into smaller subproblems.

4 Problem formulation

In this chapter, we present the mathematical formulation of the CSP. The CSP is formulated as a SCP with additional constraints. Every column of the formulation is a shift that can be carried out by one employment group, and that obeys the relevant working rules. The extra constraints ensure that the set of selected shifts fulfills the comprehensive constraints, concerning the percentage of overnight stays, regulating crew capacities, etc. The durations are given in hours, and the costs are specified in euro (per hour). First, we will introduce the sets, parameters and decision variables that are used in the formulation and throughout the rest of this paper. Next, we present the mathematical model.

Sets

D	Set of employment groups.
C	Set of crewblocks.
S	Set of candidate shifts.
L	Set of crew depot locations, relief locations at which at least one train activity starts and/or ends and locations at which at least one driving operation on which a passenger activity can be planned starts and/or ends.

For every employment group $d \in D$:

A_d	Set of all away locations for employment group d, where $A_d \subset L$.
L_d	Set of all crew depot locations where the shifts carried out by drivers of employment group d can start and/or end, where $A_d \subset L_d \subseteq L$.

Parameters

$a^1_{c,s}$	Binary parameter indicating if crewblock $c \in C$ is included in shift $s \in S$ (1) or not (0).
$a^2_{s,d}$	Binary parameter indicating if shift $s \in S$ can be allocated to employment group $d \in D$ (1) or not (0).
$a^3_{s,i}$	Binary parameter indicating if shift $s \in S$ starts at day $i \in \{1, ..., 7\}$ (1) or not (0).
$a^4_{s,l}$	Binary parameter indicating if shift $s \in S$ starts from location $l \in \bigcup_{d \in D} L_d$ (1) or not (0).
$a^5_{s,l}$	Binary parameter indicating if shift $s \in S$ ends up at location $l \in \bigcup_{d \in D} L_d$ (1) or not (0).

For every employment group $d \in D$:

n_d	Maximum number of crew rosters that can be allocated to employment group d.
γ_d	Fixed cost per week per driver of employment group d.
ζ_d	Hourly wage for a driver of employment group d.
f_d	Maximum fraction of shifts allocated to employment group d that ends with an overnight stay. Local driver are not allowed to rest at a hotel away from their home base, so that $f_d = 0$ if employment group d consists of local drivers. Otherwise, $0 \leq f_d \leq 0.5$.
$\bar{W}_d$	Maximum average number of working days per week for drivers of employment group d. This is assumed to be equal to the maximum number of shifts that can be allocated to a driver of the employment group in one week. For now, we assume that a crew roster can not contain two or more shifts that start at the same day of the week.
$\underline{V}_d$	Minimum average weekly workload per driver of employment group d.
$\bar{V}_d$	Maximum average weekly workload per driver of employment group d.
ψ_d	Hourly penalty cost incurred if the average weekly workload of a driver connected to employment group d is below $\underline{V}_d$.

For every crewblock $c \in C$:

s_c	Start time of crewblock c. No hand-over activity is required before crewblock c if the first crew requirement included in the crewblock corresponds to a start-up activity. In that case, s_c is equal to the the start time of the start-up activity. Otherwise, s_c is equal to the start time of the hand-over activity planned before crewblock c.
e_c	End time of crewblock c. No hand-over activity is required after crewblock c if the last crew requirement included in the crewblock corresponds to a shut-down activity. In that case, e_c is equal to the the end time of the shut-down activity. Otherwise, e_c is equal to the end time of the hand-over activity planned after crewblock c.
d_c	Duration of crewblock c, i.e., the time elapsed between s_c and e_c.
μ_c	Total duration of all crew requirements included in crewblock c.
u_c	Penalty cost if crewblock c is unassigned. Let θ be the hourly penalty cost for unassigned crew requirements. Then, $u_c = \theta\mu_c$.

o_c	Penalty cost if crewblock c is overcovered. Let ν be the hourly overcover cost. Then, $o_c = \nu d_c$. This cost is multiplied by the number of times that crewblock c is overcovered.

For every shift $s \in S$:

d_s	Spread time of shift s.
d_s^{break}	Minimum required meal break duration in shift s.
a_s^{night}	Binary parameter indicating if shift s is a night shift (1) or not (0).
c_s	Cost of shift s. The components of this parameter are specified in the list below.

- Salary cost. The paid time is equal to the spread time of shift s minus the minimum required break duration, i.e. $\zeta_d(d_s - d_s^{break})$.
- Night shift. The shift cost is raised with the fixed cost per night shift, c^{night}, if a_s^{night} is 1.
- Crew re-positioning. Cost of all crew repositioning operations planned in shift s.
- Non-productive cost. An additional cost of c^{npro} euro per hour is incurred for non-productive shift duration.
- Shift duration on Monday before 4 AM and on Friday after 7 AM. Employees claim an additional payment of c^{claim} euro per hour for shift duration on Monday before 4 AM and shift duration on Friday after 7 PM.
- Overnight cost. The shift cost is raised with the fixed cost $c^{overnight}$ if shift s ends at an away location of the employment group to which the shift is allocated.

<u>Decision variables</u>

Binary decision variables

U_c	Binary variable indicating if crewblock $c \in C$ is unassigned (1) or not (0). The variable is equal to 1 if crewblock c is not included in any of the selected shifts, and 0 otherwise.
X_s	Binary variable indicating if shift $s \in S$ is selected (1) or not (0).

Integer decision variables

Y_d	Number of drivers needed to carry out the shifts allocated to employment group $d \in D$.
O_c	Number of times that crewblock $c \in C$ is overcovered by the selected shifts.

Non-negative decision variable

Ψ_d	Workload allocated to employment group $d \in D$ below $\underline{V}_d Y_d$.

Mathematical formulation

The mathematical formulation of the CSP is given by (4.1) - (4.15). The dual variables of the constraints are stated in red. The dual variables of constraints (4.9) are mentioned later, as this equality constraint is first split into two inequality constraints. Let I be the set $\{1, ..., 7\}$.

$$min \quad \sum_{s \in S} c_s X_s + \sum_{c \in C} (u_c U_c + o_c O_c) + \sum_{d \in D} (\gamma_d Y_d + \psi_d \Psi_d) \tag{4.1}$$

$$s.t. \quad \sum_{s \in S} a^1_{c,s} X_s + U_c \geq 1 \qquad \forall c \in C \qquad \beta_c \tag{4.2}$$

$$\sum_{s \in S} a^1_{c,s} X_s - O_c \leq 1 \qquad \forall c \in C \qquad \delta_c \tag{4.3}$$

$$\sum_{s \in S} a^2_{s,d} a^3_{s,i} X_s \leq Y_d \qquad \forall d \in D, i \in I \qquad \epsilon_{d,i} \tag{4.4}$$

$$Y_d \leq n_d \qquad \forall d \in D \tag{4.5}$$

$$\sum_{s \in S} a^2_{s,d} X_s \leq \bar{W}_d Y_d \qquad \forall d \in D \qquad \eta_d \tag{4.6}$$

$$\sum_{s \in S} a^2_{s,d} d_s X_s + \Psi_d \geq \underline{V}_d Y_d \qquad \forall d \in D \qquad \lambda_d \tag{4.7}$$

$$\sum_{s \in S} a^2_{s,d} d_s X_s \leq \bar{V}_d Y_d \qquad \forall d \in D \qquad \kappa_d \tag{4.8}$$

$$\sum_{s \in S} a^2_{s,d} a^3_{s,i} a^5_{s,l} X_s = \sum_{s \in S} a^2_{s,d} a^3_{s,(i)(mod\ 7)+1} a^4_{s,l} X_s \qquad \forall d \in D, l \in A_d, i \in I \tag{4.9}$$

$$\sum_{s \in S} a^2_{s,d} f_d X_s \geq \sum_{s \in S} \sum_{l \in A_d} a^2_{s,d} a^5_{s,l} X_s \qquad \forall d \in D \qquad \xi_d \tag{4.10}$$

$$X_s \in \mathbb{B} \qquad \forall s \in S \tag{4.11}$$

$$U_c \in \mathbb{B} \qquad \forall c \in C \tag{4.12}$$

$$O_c \in \mathbb{Z}_{\geq 0} \qquad \forall c \in C \tag{4.13}$$

$$Y_d \in \mathbb{Z}_{\geq 0} \qquad \forall d \in D \tag{4.14}$$

$$\Psi_d \in \mathbb{R}_{\geq 0} \qquad \forall d \in D \tag{4.15}$$

The SCP is a minimization problem (4.1). Our primary main goal is to minimize the total work duration of unassigned crewblocks. Parameter θ must be large to ensure that the total duration of unassigned crew requirements is minimized. Moreover, the objective value depends on the cost of the selected shifts, the overcovered crewblocks, the shift duration and the number of drivers needed to operate the selected shifts.

In the above model, the set covering constraints (4.2) make sure that every crewblock is either covered by at least one of the selected shifts or marked as unassigned. Furthermore, constraints (4.3) determine the number of times that every crewblock in C is overcovered. Consider crewblock $c \in C$. We know that $O_c = \sum_{s \in S} a^1_{c,s} X_s - 1$ if $o_c > 0$. Furthermore, at most one of the decision variables U_c and O_c is positive for crewblock $c \in C$.

Constraints (4.4) ensure that every crew roster contains at most one shift that starts at day i, where $i \in \{1, ..., 7\}$. Furthermore, the number of crew rosters that are required to cover the shifts allocated to employment group d can not exceed the maximum number of crew rosters that can be covered by the employment group, where $d \in D$ (4.5). The number of shifts assigned to employment group d may not exceed the number of working days per week times the number of crew rosters that are required, where $d \in D$ (4.6). Constraints (4.7) calculate the total shift duration below the minimum average weekly workload. The average weekly workload of a driver may not exceed the maximum average weekly workload (4.8).

Recall that the number of overnight stays is limited to one for global drivers. Suppose that a global driver carries out a shift that starts at day i and has an overnight stay at the end, where $i \in \{1, ..., 7\}$. We assume that this shift must be succeeded by a shift that starts at day $i + 1$ and ends at the home base location of the driver. This is ensured by constraints (4.9). Constraints (4.10) make sure that the fraction of shifts with an overnight stay at the end allocated to employment group d does not exceed f_d, where $d \in D$.

In the above model, crew deadheading on train activities can occur in two ways. Firstly, every train activity (equivalently, every driving activity, as it is not useful to deadhead on a train activity that starts and ends at the same location) can be explicitly used for deadheading, e.g., if the driver does not have the required route knowledge, traction knowledge and skills to operate the corresponding crew requirement. Suppose that shift s contains a passenger activity on at least one crew requirement in crewblock c. The coefficient $a^1_{c,s}$ is equal to 0, as the employee does not perform the crew requirements in crewblock c. We should not include the cost o_c in the shift, but the cost for crew deadheading by train (c^{train} euro per hour) and the non-productive cost (c^{npro} euro per hour). If the train activity is only used for deadheading, the actual crew requirement remains unassigned. Secondly, a crewblock can be overcovered in the solution to the model. One of the drivers performs the crew requirements included in the crewblock, and the other(s) deadhead on these activities.

In this case, all corresponding $a^1_{c,s}$ are equal to 1. The penalty cost o_c is included. It is worth mentioning that o_c is larger than the cost of deadheading by train plus the non-productive cost if all train activities in crewblock c are explicitly used for crew deadheading, i.e., $o_c > (c^{train} + c^{npro})\mu_c$. We force the model to constructed shifts in which crew deadheading by train is explicitly used. The same implementation can be used if not all train activities can be used for crew deadheading. In that case, o_c is large if crewblock c contains train activities that can not be used for deadheading.

To define dual variables on constraints (4.9), we split them as follows:

$$\sum_{s \in S} a^2_{s,d} a^3_{s,i} a^5_{s,l} X_s \leq \sum_{s \in S} a^2_{s,d} a^3_{s,(i)(mod\ 7)+1} a^4_{s,l} X_s \quad \forall d \in D, l \in A_d, i \in \{1, ..., 7\} \quad \alpha^-_{d,l,i} \tag{4.9a}$$

$$\sum_{s \in S} a^2_{s,d} a^3_{s,i} a^5_{s,l} X_s \geq \sum_{s \in S} a^2_{s,d} a^3_{s,(i)(mod\ 7)+1} a^4_{s,l} X_s \quad \forall d \in D, l \in A_d, i \in \{1, ..., 7\} \quad \alpha^+_{d,l,i} \tag{4.9b}$$

Not all rostering aspects are taken into account in (4.1) - (4.15). For example, we can not guarantee that the connection time between two adjacent shifts performed by the same employee satisfies the corresponding standards without introducing a large number of additional constraints. Suppose that shift s, which starts on day $i \in Z_{>0}$, ends up at the home base location of the employment group to which the shift is allocated. We assume that shift s can be succeeded by a feasible shift that starts at day $i + j$ from the home base location of the employment group, where $j \in Z_{>0}$. If shift s ends up at an away location of the employment group, we assume that it can be succeeded by a feasible shift that starts at day $i + 1$ from the away location and ends at the home base location of the group. Problems can arise during the crew rostering phase. For example, it is possible that more drivers are needed to cover the shifts allocated to an employment group than the number of drivers determined in the crew scheduling phase. In that case, the cost of the final solution increases. On the other hand, the cost of the solution can decrease if two shifts that start on the same day can be assigned to the same driver, for example.

5 Methodology

The methodology used to solve the CSP is outlined in this chapter. The solution method is decomposed into four stages. In the first stage, workblocks are constructed. A workblock is the sequence of crew requirements between two relief opportunities. In the second stage, multiple workblocks are merged into one crewblock to reduce the problem size. The activities included in one crewblock are performed by the same driver in sequence. In the third stage, the required hand-over activities are planned between the crewblocks. The hand-over activity planned between two crewblocks is redundant if these crewblocks are covered by the same driver. In the final stage, column generation is used to solve the CSP. The restricted master problem (RMP) is the LP relaxation of (4.1) - (4.15), and the pricing problem is formulated as a resource constrained shortest path (RCSP) algorithm. The size of (4.1) - (4.15) grows exponentially with the number of crewblocks and employment groups, and explicit enumeration of all feasible shifts is not possible within a reasonable time limit for real-life problem instances. The advantage of column generation is that this solution method does not require the explicit consideration of all variables.

The remainder of this chapter is organized as follows. Section 5.1 describes three methods that can be used to construct crewblocks, given the set of workblocks. The goal of Section 5.2 is to describe the approach used to plan the hand-over activities between the crewblocks. The column generation technique is described in Section 5.3.

5.1 Crewblock construction

The intention of this section is to provide three methods that are used to combine multiple workblocks into one crewblock. The advantage of a CSP in which multiple workblocks are merged into one crewblock is its reduced size, such that the problem is easier and faster to solve (Abbink et al. (2011)). There is a trade-off between the duration of the crewblocks and the cost of the solution. In particular, one might be able to reduce the cost of the solution by decreasing the number of workblocks that are combined into one crewblock.

The method described in Subsection 5.1.1 starts with crewblocks that consist of all crew requirements planned in sequence on a mobilized locomotive. Thereafter, the method splits the crewblocks, based on cutting rules. Each workblock is included completely in one crewblock. The method described in Subsection 5.1.2 is based on block cutting heuristics and constructs all possible crewblocks, given some predefined parameters. The problem to select a subset of crewblocks such that every workblock is included in exactly one selected crewblock is formulated as a SPP. The method described in Subsection 5.1.3 merges workblocks based on predefined parameters, concerning the minimum and maximum length of a crewblock, and the idle time between two crew requirements included in the same crewblock.

5.1.1 Split large crewblocks based on cutting rules

It is likely that the impact of delay on the vehicle schedule and crew schedule increases with the number of crew changes on a mobilized locomotive. Suppose that one train is delayed and that a crew change on the locomotive is planned. The driver who takes over the locomotive has to wait for the train to arrive and it is reasonable that both drivers involved encounter problems during the rest of their working day. Reserve crew is needed to cover the crew requirements that remain unassigned as a result of disruption during daily operations. Reserve crew is an expensive resource. The method described in this subsection is stated on this observation.

The set of crew requirements derived from the train activities planned in the model week is given by R. Locline cycle $l \in LC$ is represented by the sequence of crew requirements that correspond with the train activities planned in time behind each other on cycle l, i.e. $l = \{r_{1,l}, ..., r_{n_l,l}\}$ where $r_{i,l} \in R$ for all positions $i \in \{1, ..., n_l\}$ and n_l the number of train activities in locline cycle l. Suppose that $r_{i,l}$ corresponds with a shut-down activity, where $i \in \{1, ..., n_l\}$. Crew requirement $r_{(i)(mod\ n_l)+1,l}$ must correspond with a start-up activity. In general, the idle time between these two activities is relatively large and it is unproductive to include them in the same crewblock. One crewblock is constructed for the sequence of crew requirements planned on a mobilized locomotive. Every crewblock starts with a start-up activity and contains all crew requirements up to and including the next planned shut-down activity. The duration of the individual crewblocks might be large, which makes it difficult to construct feasible shifts covering every crewblock at least once. For this reason, the individual crewblocks are cut into smaller crewblocks. The cutting criterion is described below.

Consider crewblock $c \in C$. This crewblock is represented by the sequence $(r_{1,c}, ..., r_{n_c,c})$, where $r_{i,c} \in R$ for all positions $i \in \{1, ..., n_c\}$ and n_c is the number of crew requirements included in crewblock c. Let R_c be the set of crew requirements in c after which a relief opportunity exists. The last crew requirement in c is not included in R_c, i.e $R_c \subseteq \{r_{1,c}, ..., r_{n_c-1,c}\}$. Binary parameter $a_{d,c}$ indicates if employment group $d \in D$ has the traction knowledge, route knowledge and skills to operate the crew requirements in crewblock $c \in C$ (1) or not (0). The maximum spread time of national and international shifts operated by employment group $d \in D$ are given by $\bar{d}_d^{nat}$ and $\bar{d}_d^{int}$, respectively. Introduce the parameter $\bar{d}_{d,c}$, for all $d \in D$ and $c \in C$. The parameter is equal to $\bar{d}_d^{int}$ if crewblock $c \in C$ contains at least one crew requirement that starts or ends abroad for employment group $d \in D$. Otherwise, $\bar{d}_{d,c}$ is the maximum of $\bar{d}_d^{nat}$ and $\bar{d}_d^{int}$.

Crewblock $c \in C$ is split into crewblocks c_1 and c_2 if the number of crew requirements in R_c is larger than 0 and if $2d_c$ is larger than the duration of the maximum spread time of the most restricted employment group that can cover the crewblock, i.e. $2d_c > \min_{e \in \{d \in D | a_{c,d}=1\}}(\bar{d}_{e,c})$. The cutting rule claims that the crewblock is split such that $|d_{c_1} - d_{c_2}|$ is minimized, given that the last crew requirement in c_1 is an element from the set R_c. $|x|$ is the absolute value of x, i.e. $|x| = x$ if $x \geq 0$ and $|x| = -x$ if $x < 0$. Relevant parameters of c_1 and c_2 are determined, e.g., the start time and end time of the crewblocks. Furthermore, C is updated. The algorithm terminates if none of the crewblocks needs any more splitting or if $|R_c| = 0$ for every crewblock $c \in C$ for which $2d_c > \min_{e \in \{d \in D | a_{c,d}=1\}}(\bar{d}_{e,c})$.

5.1.2 Block cutting and set partitioning

The number of relief opportunities increases with the number of crew requirements, the number of relief locations and the idle time between two train activities planned in sequence on the same locomotive. From a management point of view, it is inefficient and sometimes even undesirable to change crew after a small period of time. On the other hand, the companies' employees request enough variation between the activities included in a shift, see the real-life case of NS reizigers described in the introduction. The crewblock construction method described in this subsection is able to take these conflicting interests into account.

Two crew requirements planned in sequence on a locomotive must be included in the same crewblock if the idle time between the activities is smaller than $\underline{M}$. This is an *if*-statement, not an *if-and-only-if*-statement. Furthermore, the non-productive time of employees is reduced by specifying the maximum idle time between two consecutive activities included in the same crewblock ($\bar{M}$). Two crew requirement planned in sequence on a locomotive are included in different crewblocks if the idle time between the activities is larger than $\bar{M}$ and if a relief opportunity exists between the activities. It must hold that $\underline{M} < \bar{M}$.

Recall that crewblock $c \in C$ is represented by the sequence $(r_{1,c}, ..., r_{n_c,c})$, with $r_{i,c} \in R$ for all $i \in \{1, ..., n_c\}$. Let δ_c be the time elapsed between the start of crew requirement $r_{1,c}$ and the end of crew requirement $r_{n_c,c}$, for all $c \in C$. We want to enforce the value of δ_c to be in the interval $[\underline{T}, \bar{T})$. In practice, it is not necessarily possible that $\underline{T} \leq \delta_c < \bar{T}$ for all $c \in C$. Let $C_x \subseteq C$ be the set of crewblocks for which δ_c is at least x hours. We want to minimize $|C_{\bar{T}}|$, since small crewblocks are in general easier to cover than large crewblocks.

The method described in this subsection uses the idea of block cutting heuristics to cut locline cycles into crewblocks, given a set of cutting patterns. The method is decomposed into two phases, namely block cutting and block selecting. The block cutting process cuts the locline cycles only at relief locations, given that the idle time between the activities is greater than or equal to $\underline{M}$. There are different cutting patterns, as the locline cycles can be cut at several points. Recall that two crew requirements planned in sequence on the same locline cycle are not included in the same crewblock if the idle time between the activities is larger than $\bar{M}$ and if there is a relief opportunity between the two crew requirements. Furthermore, two or more workblocks can only be combined into one crewblock if the corresponding δ_c is smaller than or equal to $\bar{T}$. All feasible crewblocks are generated given the set of cutting patterns, and a large candidate list of crewblocks is formed which covers every workblock at least once ($\bar{C}$), see example 5.1.A.

Example 5.1.A: Parameter $\underline{M}$ is equal to the hand-over duration of 1 minute, and $\bar{M}$ is 2 hours. Furthermore, we want to enforce that δ_c is in the interval [2,6], for all $c \in C$. Consider the following part of a locline cycle.

$$\begin{tikzcd}
0030 \arrow[r, dashed] & \overset{w_1}{0230\text{-}0430} \arrow[r] & \overset{w_2}{0450\text{-}0620} \arrow[r] & \overset{w_3}{0630\text{-}08300} \arrow[r] & \overset{w_4}{1100\text{-}1900} \arrow[r] & \overset{w_5}{1910\text{-}2100} \arrow[r, dashed] & 2310
\end{tikzcd}$$

Let W be the set of workblocks constructed in the first stage of the solution process. In this case, $W = \{\cdots, w_1, ..., w_5, \cdots\}$. Crewblock $c \in C$ is represented by the sequence of workblocks included in c, i.e. $c = (w_{1,c}, ..., w_{m_c,c})$ where m_c is the number of workblocks included in crewblock c and $w_{i,c} \in W$ for all positions $i \in \{1, ..., m_c\}$. The candidate list of crewblocks is $\bar{C} = \{\cdots, (w_1), (w_2), (w_3), (w_4), (w_5), (w_1, w_2), (w_2, w_3), (w_1, w_2, w_3), \cdots\}$, given that every workblock can be carried out by the same subset of employment groups.

The LP relaxation of (4.1) - (4.15) is the master problem of the column generation method used in this paper. Every workblock is included in exactly one crewblock in C in (4.1) - (4.15). The problem to select elements in $\bar{C}$ such that every workblock is included in exactly one selected crewblock is formulated as a SPP. Our primary goal is to minimize $|C_{\bar{T}}|$. Our secondary goal is to minimize $|C - C_{\underline{T}}|$. Our third goal is to maximize $|C|$. The multi-objective problem is solved by combining the multiple objectives into a single-objective scalar function.

Binary parameter $a^6_{w,c}$ indicates if workblock $w \in W$ is included in crewblock $c \in \bar{C}$ (1) or not (0). The indicator function I_x is equal to 1 if x holds, and 0 otherwise. We can formulate the problem to select crewblocks from the candidate list of crewblocks using binary variables x_c indicating if crewblock $c \in \bar{C}$ is selected (1) or not (0). Every workblock must be included in exactly one of the selected crewblocks, and the problem is formulated as a SPP.

$$\begin{aligned} min \quad & \alpha_1 \sum_{c\in\bar{C}} I_{(\delta_c > \bar{T})} x_c + \alpha_2 \sum_{c\in\bar{C}} I_{(\delta_c < \underline{T})} x_c - \alpha_3 \sum_{c\in\bar{C}} x_c && && (5.1)\\ s.t. \quad & \sum_{c\in\bar{C}} a^6_{w,c} x_c = 1 && \forall w \in W && (5.2)\\ & x_c \in \mathbb{B} && \forall c \in \bar{C} && (5.3) \end{aligned}$$

The parameters α_1, α_2 and α_3 are chosen such that our goals are taken into account, i.e. $\alpha_1 \gg \alpha_2 \gg \alpha_3$. Constraints (5.2) ensure that every workblock is included in exactly one crewblock in C, where $C = \{c \in \bar{C} | x_c = 1\}$. It is likely that the optimal solution of (5.1) - (5.3) is not unique, see example 5.1.B.

Example 5.1.B: W and $\bar{C}$ are described in example 5.1.A. The following solutions fulfill the constraints in (5.1) - (5.3).

1. $C = \{(w_1), (w_2), (w_3), (w_4), (w_5)\}$
2. $C = \{(w_1, w_2), (w_3), (w_4), (w_5)\}$
3. $C = \{(w_1), (w_2, w_3), (w_4), (w_5)\}$
4. $C = \{(w_1, w_2, w_3), (w_4), (w_5)\}$

Examine that the objective function of (5.1) - (5.3) is minimized under solution 2 or 3. As a result, the optimal solution of (5.1) - (5.3) is not unique.

In this paper, we will investigate the effect of different implementation options on the performance of the column generation technique described in Section 5.3. We can only compare the results obtained with different implementation options if the set of crewblocks added to (4.1) - (4.15) is the same in each run, for given values of $\underline{M}$, $\bar{M}$, $\underline{T}$ and $\bar{T}$. We have seen before that the optimal solution of (5.1) - (5.3) is not necessarily unique. One way to solve this problem is to select $C \subseteq \bar{C}$ once, and use the resulting set of crewblocks in each independent run.

5.1.3 Merge multiple workblocks into one crewblock to reduce the problem size

The method described in this subsection is a heuristic version of the construction method described in Subsection 5.1.2. This method does not construct the large candidate list of crewblocks. Multiple workblocks are combined into one crewblock, given the different objectives and parameter settings. A local search algorithm is added to improve the solution of the heuristic. Different neighboring solutions are considered to improve the initial solution.

We will now describe the process used to construct the initial solution. Our primary goal is to minimize $|C_{\bar{T}}|$. Our secondary goal is to minimize $|C - C_{\underline{T}}|$. Finally, we want to maximize the number of crewblocks in C. An initial solution is constructed by combining crewblocks in $C - C_{\underline{T}}$ with their adjacent crewblocks, without increasing the number of elements in $C_{\bar{T}}$. Let H be the set of elements for which δ_c is at least $\underline{T}$, or that can not be combined with one of its adjacent crewblocks. Initially, H equals $C_{\underline{T}}$. The algorithm selects crewblock $c = \text{argmin}_{\gamma \in C-H}(\delta_\gamma)$. Let c_c^- and c_c^+ be the crewblocks scheduled directly before and after crewblock c, respectively. The selected crewblock is combined with the crewblock $c' \in \{c_c^-, c_c^+\}$ for which the following conditions are fulfilled, with ties broken by smallest idle time.

1. The idle time between c and c' is smaller than or equal to $\bar{M}$.
2. At least one employment group has the qualifications and licensing for all crew requirement included in c and c', i.e., $max_{d \in D}(a_{d,c} + a_{d,c'}) = 2$.
3. The number of elements in $C_{\bar{T}}$ does not increase by combining c and c'.

Crewblock c can not be combined with one of its adjacent crewblocks if not all conditions 1 - 3 are valid for both c_c^- and c_c^+. As a result, c must be added to H. Otherwise, C and H are updated. The process is repeated until $C = H$.

The local search heuristic tries to improve the initial solution, by reducing the number of elements in $C - C_{\underline{T}}$ without increasing $|C_{\bar{T}}|$. Note that $|C_{\bar{T}}|$ can not be reduced, given the rules used by constructing the initial solution. The local search heuristic is applied to every locline cycle separately. Consider locline cycle $l \in LC$. The set of crewblocks planned on this locline cycle is given by C_l. Workblocks are swapped between crewblocks if that (potentially) leads to a reduction of $|C - C_{\underline{T}}|$. The improvement heuristic is terminated if the number of iterations exceeds the user-specified number of iterations or if $|C_l| = 1$.

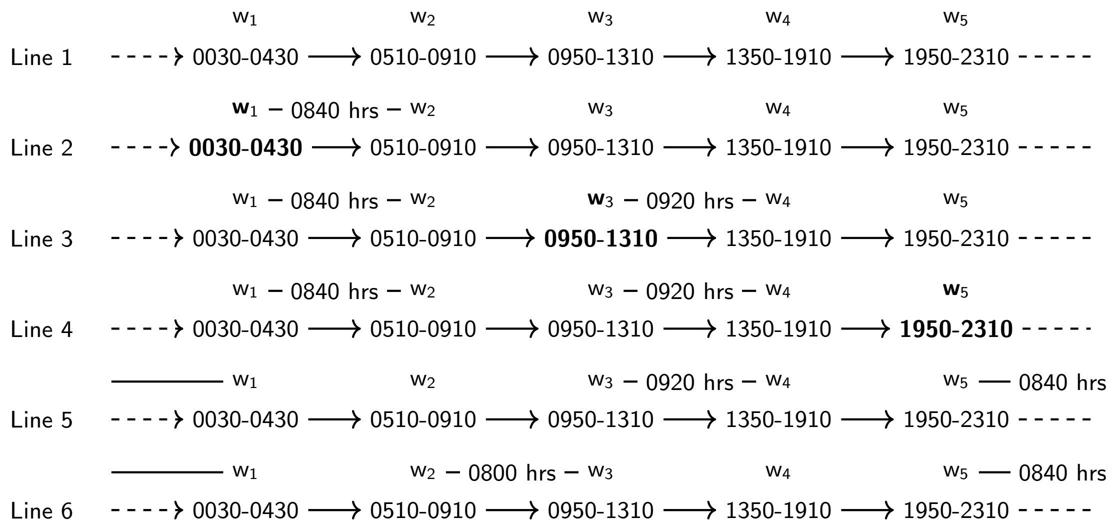
In each iteration, the algorithm randomly selects one element from the set C_l, say crewblock c. A second crewblock (c') is randomly selected from the set $\{c_c^-, c_c^+\}$. Recall that crewblock $\gamma \in C$ is represented by the sequence of workblocks in γ, i.e. $\gamma = (w_{1,\gamma}, ..., w_{m_\gamma,\gamma})$ where $w_{i,\gamma} \in W$ for all positions $i \in \{1, ..., m_\gamma\}$ and m_γ the number of workblocks in γ. The algorithm selects workblock $w \in c'$ adjacent to crewblock c. That is, $w = w_{m_{c'},c'}$ if $c' = c_c^-$ and $w = w_{1,c'}$ if $c' = c_c^+$. Let c_w be the crewblock that is represented by the sequence $(w, w_{1,c}, ..., w_{m_c,c})$ if $c' = c_c^-$ and $(w_{1,c}, ..., w_{m_c,c}, w)$ if $c' = c_c^+$. Crewblock c'_w is represented by the sequence $(w_{1,b}, ..., w_{m_b-1,b})$ if $c' = c_c^-$ and $(w_{2,b}, ..., w_{m_b,b})$ if $c' = c_c^+$. Please note that $m_{c'_w}$ is equal to zero if $m_{c'}$ is one. If none of the conditions listed below is fulfilled, crewblocks c and c' are replaced by crewblocks c_w and c'_w. Otherwise, C_l is not updated. The iteration counter is increased by one. This process is repeated until the stopping criterion is fulfilled.

1. There is no employment group that has the route knowledge, traction knowledge and skills for all crew requirements in crewblock c_w, i.e., $a_{d,c_w} = 0$ for all $d \in D$.

2. The idle time between c and c' exceeds $\bar{M}$.
3. $\delta_{c_w} > \bar{T}$.
4. The number of crewblocks in $C_{\underline{T}}$ decreases due to the interchange of workblock w, i.e., $I_{\delta_c \geq \underline{T}} + I_{\delta_{c'} \geq \underline{T}} > I_{\delta_{c_w} \geq \underline{T}} + I_{\delta_{c'_w} \geq \underline{T}}$. I_x indicates of condition x holds (1) or not (0). If $m_{c'_w} = 0$, then $I_{\delta_{c'_w} \geq \underline{T}} = 0$.
5. The number of crewblocks in $C_{\underline{T}}$ does not change due to the interchange of workblock w, crewblock c'_w is non-empty, and $\delta_c + \delta_{c'} < \delta_{c_w} + \delta_{c'_w}$ OR $|\delta_c - \delta_{c'}| < |\delta_{c_w} - \delta_{c'_w}|$.

Figure 3 is introduced to clarify the approach described in this subsection. In this example, a locline is a sequence of crew requirements planned on one day for which the same locomotive is used. No other loclines are planned in time behind this locline and the corresponding locline cycle is a circulation of only one locline. $\underline{T}$ is 5 hours and 20 minutes and $\bar{T}$ is 9 hours and 20 minutes. Further, $\bar{M}$ is 50 minutes. There is at least one employment group that has the ability to execute all crew requirements that correspond with the train activities planned on the locline given in Figure 3.

Figure 3: Example: Initial solution and local search heuristic


w1 w2 w3 w4 w5
Line 1 0030-0430 0510-0910 0950-1310 1350-1910 1950-2310
w1 – 0840 hrs – w2 w3 w4 w5
Line 2 0030-0430 0510-0910 0950-1310 1350-1910 1950-2310
w1 – 0840 hrs – w2 w3 – 0920 hrs – w4 w5
Line 3 0030-0430 0510-0910 0950-1310 1350-1910 1950-2310
w1 – 0840 hrs – w2 w3 – 0920 hrs – w4 w5
Line 4 0030-0430 0510-0910 0950-1310 1350-1910 1950-2310
w1 w2 w3 – 0920 hrs – w4 w5 — 0840 hrs
Line 5 0030-0430 0510-0910 0950-1310 1350-1910 1950-2310
w1 w2 – 0800 hrs – w3 w4 w5 — 0840 hrs
Line 6 0030-0430 0510-0910 0950-1310 1350-1910 1950-2310

Workblocks w_1 - w_5 are constructed in the first step. Line 2 - 4 give a schematic representation of the construction of the initial solution, where the bold block is considered. Line 5 - 6 represent the local search heuristic. In the end, $\underline{T} \leq \delta_c < \bar{T}$ for all $c \in C$.

5.2 Hand-over planning method

The hand-over of responsibility between the drivers involved in a crew change on a mobilized locomotive is planned during or directly after the crewblock construction phase. The advantage is the reduced size of the CSP, as the number of feasible shifts might be reduced. Moreover, column generation techniques can not be used if the feasibility of a variable depends on the selection of other variables.

In general, a shut-down activity is planned in the vehicle schedule if the idle time between two activities planned on a mobilized locomotive is relatively large. No hand-over of responsibilities is required after a shut-down activity. Suppose that the last crew requirement in crewblock $c \in C$ is not a shut-down activity. A hand-over of responsibility is required if a driver supplants the driver that has operated the locomotive and performed the crew requirements in c. The crewblock that is planned in sequence on the same locomotive as crewblock c is c_c^+. A window of relief opportunities exists if the time gap between crew requirements $r_{n_c,c}$ and r_{1,c_c^+} is larger than the hand-over duration of 1 minute. The mobilized locomotive may not be left unattended, and it is not of much interest which crew member has some idle time on the locomotive. The start time of the hand-over activity is determined as follows.

- The hand-over activity is planned directly after $r_{n_c,c}$ with a chance of ρ_1 percent.
- The hand-over activity is planned directly before r_{1,c_c^+} with a chance of ρ_2 percent.
- The hand-over activity is planned randomly between the end of $r_{n_c,c}$ and the start of r_{1,c_c^+} with a chance of ρ_3 percent. Each hand-over activity starts at a time point that is rounded to whole minutes.

The total percentage adds up to 100 and the percentages are non-negative, i.e. $\rho_1+\rho_2+\rho_3 = 100$ and $\rho_1, \rho_2, \rho_3 \geq 0$. It is worth mentioning that no hand-over of responsibility is required between crewblocks c and c_c^+ if these crewblocks are included in the same shift. The following example is used to clarify the hand-over planning method.

Example: Assume that the end time of the last activity in the considered crewblock is 14:00 and that the next crew requirement planned on the same locomotive starts at 14:30. With a chance of ρ_1 percent, the hand-over activity is planned from 14:00 to 14:01. With a chance of ρ_2 percent, the hand-over activity takes place from 14:29 to 14:30. The start time of the hand-over activity is a random time point in the interval [14:00 14:29] with a chance of ρ_3 percent. Recall that the start time is rounded to whole minutes.

5.3 Column generation

Recall that the CSP is NP-complete. In practice, complete enumeration of all feasible shifts is computationally intractable for large problem instances. Column generation techniques have been successfully applied to real-life crew scheduling problems, see Chapter 3. This solution method does not require the explicit construction of all variables. The problem is decomposed into a restricted master problem (RMP) and a pricing problem. The general idea of column generation is given in Subsection 5.3.1. The RMP and the pricing problem that are used in this paper are described in Subsection 5.3.2 and Subsection 5.3.3, respectively.

5.3.1 General idea of column generation

Column generation techniques can be used to solve (integer) linear programs that are too large to consider all variables explicitly. Column generation exploits the idea that most of the variables are non-basic in the optimal solution of the (integer) linear program. The problem is split into two subproblems, namely the RMP and the pricing problem. The RMP is the (linear programming (LP) relaxation of the integer) linear program in which only a subset of variables is considered. Variables that have the potential to improve the objective function of the RMP and the original program are generated in the pricing problem.

The process of column generation works as follows. An initial set of variables is generated and added to the RMP, such that a solution exists. For example, a variable is generated for every crewblock representing an infeasible shift that covers that crewblock. The cost of these shifts is very high so that they are only included in the solution if there is no feasible way of covering the crewblock. The RMP is solved with the current set of variables and the dual prices are obtained for every constraint in the RMP. We assume without loss of generality that the RMP is a minimization problem. The objective function of the pricing problem is the reduced cost of the solution variable with respect to the current dual prices. We want to find a feasible variable with minimal reduced cost. The constraints of the pricing problem ensure that the generated variables obey the naturally occurring constraints. The pricing problem is solved. A variable with negative reduced cost is identified if the objective value of the pricing problem is negative. The variable has the potential to improve the objective function of the RMP, and is therefore added to the subset of variables. The RMP is re-solved and the dual prices are updated. The pricing problem is solved with the new dual prices. The optimal solution of the linear program is found if no variable with negative reduced cost can be identified in the pricing problem.

5.3.2 Restricted master problem

The mathematical model presented in Chapter 4 describes the CSP at the medium-size European rail-cargo company. The original program is given by (4.1) - (4.15). The master problem is the LP relaxation of the original program, i.e., the LP obtained by dropping the integrality constraints (4.11) - (4.14). The RMP is the master problem in which only a limited number of variables is considered. The dual variables indicate which constraints of the LP relaxation of (4.1) - (4.15) are hardest to satisfy. The dual variables of the constraints are stated in Chapter 4. The optimal solution of the master problem is found if no variable with negative reduced cost can be identified in the pricing problem. The optimal solution of the master problem provides a lower bound on the optimal solution of the original program. The algorithm can not guarantee that the optimal solution of the original program is found as well, unless the integrality constraints (4.11) - (4.14) are fulfilled by the LP solution. It is likely that slightly different variables are required by the original program.

5.3.3 Pricing problem

Variables that have the potential to improve the objective function of the RMP (and also of the original program) are identified in the pricing problem. The objective function of this problem is the reduced cost of the solution variable with respect to the current dual values of the constraints in the RMP. Recall that the RMP is a minimization problem. As a result, the objective function of the pricing problem must be minimized. The reduced cost of shift s is calculated using Equation (5.4).

$$\begin{aligned}
r_s = {} & c_s + \sum_{c \in C} a^1_{c,s} \beta_c - \sum_{c \in C} a^1_{c,s} \delta_c - \sum_{d \in D} a^2_{s,d} \sum_{i=1}^{7} a^3_{s,i} \epsilon_{d,i} - \sum_{d \in D} a^2_{s,d} \eta_d \\
& + \sum_{d \in D} a^2_{s,d} d_s \lambda_d - \sum_{d \in D} a^2_{s,d} d_s \kappa_d - \sum_{d \in D} a^2_{s,d} \sum_{l \in A_d} \sum_{i=1}^{7} \left(a^3_{s,i} a^5_{s,l} - a^3_{s,(i \ mod \ 7)+1} a^4_{s,l} \right) \alpha^-_{d,l,i} \\
& + \sum_{d \in D} a^2_{s,d} \sum_{l \in A_d} \sum_{i=1}^{7} \left(a^3_{s,i} a^5_{s,l} - a^3_{s,(i \ mod \ 7)+1} a^4_{s,l} \right) \alpha^+_{d,l,i} - \sum_{d \in D} a^2_{s,d} \Big(\sum_{l \in A_d} a^5_{s,l} - f_d \Big) \xi_d \\
= {} & c_s + \sum_{c \in C} a^1_{c,s} \left(\beta_c - \delta_c \right) + \sum_{d \in D} a^2_{s,d} \Bigg(\Big(- \sum_{i=1}^{7} a^3_{s,i} \epsilon_{d,i} \Big) - \eta_d + d_s (\lambda_d - \kappa_d) \\
& - \Big(\sum_{l \in A_d} \sum_{i=1}^{7} (a^3_{s,i} a^5_{s,l} - a^3_{s,(i \ mod \ 7)+1} a^4_{s,l}) (\alpha^-_{d,l,i} - \alpha^+_{d,l,i}) \Big) - \Big(\sum_{l \in A_d} a^5_{s,l} - f_d \Big) \xi_d \Bigg)
\end{aligned} \tag{5.4}$$

The remainder of this subsection is organized as follows. We formulate the pricing problem as a resource constrained shortest path (RCSP) algorithm. General information about this algorithm is given in Paragraph 5.3.3.1. Dominance rules are used to limit the number of paths that must be researched by the pricing problem. Two commonly used dominance rules are described in Paragraph 5.3.3.2, and an example is used to discuss the difficulties encountered with each of these rules. The paths that are generated inside the pricing problem must obey the naturally occurring constraints. This can only be assured if the pricing problem is split by employment group, nationality and valid break requirements. This is described in Paragraph 5.3.3.3. We also describe the idea to split the resulting pricing subproblems on weekday in Paragraph 5.3.3.3. Abbink et al. (2011) show that this results in significant performance improvements. The independent pricing subproblems can be solved in parallel, and the number of threads depends on the number of CPU cores available. Each pricing subproblem is stated on an acyclic directed graph. These graphs are described in detail in Paragraph 5.3.3.4.

5.3.3.1 Resource constrained shortest path algorithm

The RCSP problem is the problem of finding a least cost path between two nodes in an acyclic network obeying a set of resource constraints. The RCSP algorithm is stated on an acyclic directed graph $G = (V, A)$. We will now describe briefly how this graph is related to the CSP. Detailed information about the nodes, arcs and resources can be found in Paragraph 5.3.3.4.

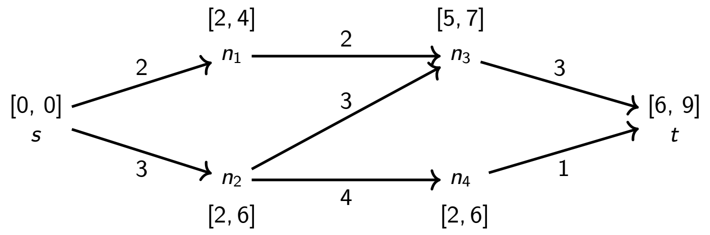
The RCSP problem asks for the computation of a resource feasible path from the source to the target with minimal reduced cost. Two nodes are constructed for each location at which a log in and log out activity can be planned. These nodes determine the start and end location of the path. Two additional nodes are constructed to offset the night cost and driving time regulations. The other nodes in V represent the start and/or end of a crewblock or possible passenger activity. These nodes are associated with a time point and location. Crewblocks and possible passenger activities are represented by the arc between the nodes corresponding with the start and end of the event. An arc can also include crew deadheading trips, idle time and/or meal breaks. For example, a meal break can be planned between two activities included in the same crewblock.

A path $P = (v_0, ..., v_p)$ is a finite sequence of nodes for which it holds that $(v_i, v_{i+1}) \in A$ for all positions $i = 0, 1, ..., p - 1$. The objective is to select a path in G from the source to the target with minimal cost that is resource feasible. Resources are included to ensure that the solution variables satisfy the relevant operational constraints. In our case, the resources keep track of the reduced cost, duration, driving time, meal break duration, nationality and night time spend.

Resource constraints are formulated by means of resource consumptions on the arcs and resource windows on the nodes. Let N be the number of resources. The resource vector is given by $B =$

$(B^1, ..., B^N)^T \in \mathbb{R}^N$, and the elements of this vector are the resource variables. The resource windows associated with node $i \in V$ are denoted by $[l_i, u_i]$, where $l_i, u_i \in \mathbb{R}^N$ and $l_i \leq u_i$. The changes in the resource consumption associated with the arc $(i, j) \in A$ are given by the vector $f_{i,j} = (f^r_{i,j})^N_{r=1}$ of resource extension functions (REFs). The REF $f_{i,j}$ depends on a resource vector $B_i \in \mathbb{R}^N$. Resource vector B_i is the resource consumption accumulated along a path from the source up to node i. The result $f_{i,j}(B_i) \in \mathbb{R}^N$ is the resource consumption accumulated along a path for the source to node j via node i. In our case, the algorithm is modeled such that there is no interdependence between the different resources and $f_{i,j}(B_i) = B_i + b_{i,j}$. The elements of the vector $b_{i,j}$ are the resource consumptions on arc (i, j). Path $P = (v_0, ..., v_p)$ is feasible with respect to the resources if there exist resource vectors $B_i \in [l_{v_i}, u_{v_i}]$ for all positions $i = 0, 1, ..., p$ such that $f_{v_i,v_{i+1}}(B_i) = B_{i+1}$ holds for all $i = 0, 1, ..., p - 1$.

Example: Let us consider the following acyclic directed graph with six nodes, seven arcs, and one resource.



The resource windows and the resource consumptions are given. The number of paths from the source (s) to the target (t) is three, namely $P_1 = (s, n_1, n_3, t)$, $P_2 = (s, n_2, n_3, t)$ and $P_3 = (s, n_2, n_4, t)$. We will now examine whether or not these paths are resource feasible.

P_1: $B_s = 0 \in [0, 0]$, $B_{n_1} = 0 + 2 = 2 \in [2, 4]$, $B_{n_3} = 2 + 2 = 4 \notin [5, 7]$.

P_2: $B_s = 0 \in [0, 0]$, $B_{n_2} = 0 + 3 = 3 \in [2, 4]$, $B_{n_3} = 3 + 3 = 6 \in [5, 7]$, $B_t = 6 + 3 = 9 \in [6, 9]$.

P_3: $B_s = 0 \in [0, 0]$, $B_{n_2} = 0 + 3 = 3 \in [2, 4]$, $B_{n_4} = 3 + 4 = 7 \notin [2, 6]$.

Path P_2 is the only path in the acyclic directed graph that is resource feasible.

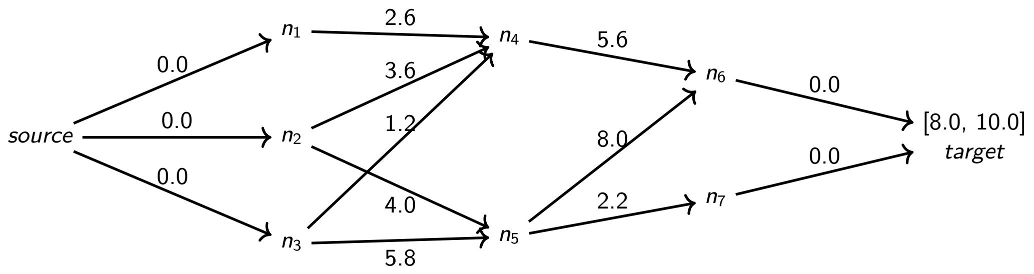
5.3.3.2 Dominance rules

The RCSP algorithm is potentially very slow when all paths in the graph must be considered. The algorithm uses the concept of dominance rules to restrict the number of paths that must be researched. First, we describe how dominance rules are being used to restrict the number of paths to be considered. Second, two commonly used dominance rules are described and an example is used to discuss the difficulties encountered with each of these rules.

All dominance rules specify resource bounds on each node, given the resource windows on the nodes and resource consumptions associated with the arcs. These bounds are different from the resource windows we defined before (the $[l_i, u_i]$ on node $i \in V$). The resource bounds on node $i \in V$ are denoted by $[a_i, b_i]$, where $a_i, b_i \in \mathbb{R}^N$ and $l_i \leq a_i \leq b_i \leq u_i$. A path from the source to node i is pruned if the corresponding resource vector B_i is not in the interval $[a_i, b_i]$. Recall that N resources are added to the labels. The RCSP problem asks for the computation of a path with minimal cost obeying a set of resource constraints, assuming without loss of generality that the pricing problem is a minimization problem. Suppose that the first resource value must be minimized at the target. The levels of the other resources do not play a role in the optimization, and it must only hold that the path is feasible with respect to these resources. Consider the resource vectors $B_i \in [a_i, b_i]$ and $B_i^{'} \in [a_i, b_i]$. The resource vector B_i' is dominated by the resource vector B_i at node i if $B_i^1 \leq B_i^{'1}$. The path corresponding with B_i' is pruned. Some paths are only dominated at the target.

Consider the acyclic directed graph shown in Figure 4 with nine nodes, thirteen arcs, and one resource. The resource consumption associated with each arc is specified. We require the resource value to be between 8.0 and 10.0 on the target, and leave it free on the other nodes (equivalently, in [0.0, 10.0] on the other nodes, as the resource consumptions are non-negative). Our goal is to find a path from the source to the target with minimal resource value.

Figure 4: Acyclic directed graph with nine nodes, thirteen arcs and one resource



There are a number of illegal paths, e.g.,

$$source \to n_2 \to n_5 \to n_7 \to target,\ B_{target} = 6.2.$$

$$source \to n_3 \to n_5 \to n_6 \to target,\ B_{target} = 13.8.$$

The RCSP algorithm is potentially very slow if no dominance rule is set. The path from the source to node i corresponding with $B_i \in \mathbb{R}^N$ is only pruned if $B_i \notin [l_i, u_i]$. For the example, the algorithm produces the following paths.

$$source \to n_1 \to n_4 \to n_6 \to target,\ B_{target} = 8.2.$$

$$source \to n_2 \to n_4 \to n_6 \to target,\ B_{target} = 9.2.$$

$$source \to n_3 \to n_5 \to n_7 \to target,\ B_{target} = 8.0.$$

Especially for large RCSP problems, it is recommended to set a maximum duration, or to specify the number of feasible paths after which the algorithm terminates. Unfortunately, it is likely this approach will not find (near-)optimal solutions.

Consider the graph shown in Figure 4. If we only consider the path from the source to node i for which $B_i \in \mathbb{R}$ is minimal, it might occur that no feasible path is found even though resource feasible paths from the source to the target exist. In terms of the dominance that we described, this corresponds to the case where $[a_i, b_i] = [l_i, u_i]$ for all $i \in V$. Only a small fraction of paths is considered. The best resource value in n_4 is 1.2 and the best resource value in n_5 is 4.0. The choice at the target is between 6.8 and 6.2. Both values are outside the resource window at the target and no feasible paths are found. We have seen before that three resource feasible paths exist.

The **heuristic dominance** rule uses pre-processing to specify resource bounds on each node. The graph is traversed forwards once, and backwards once, and at each node the *feasible bounds* are stored. The feasible bounds are such that a resource vector must be within these bounds in order to be able to extend to a feasible path. This is not an *if-and-only-if statement*, and it might occur that the optimal path is not found.

We will now describe how the feasible bounds are computed in general. Recall that the resource windows associated with node $i \in V$ are denoted by $[l_i, u_i]$, with $l_i, u_i \in \mathbb{R}^N$ and $l_i \leq u_i$. The feasible bounds at node i are given by $[a_i, b_i]$, where $a_i, b_i \in \mathbb{R}^N$ and $l_i \leq a_i \leq b_i \leq u_i$. The graph is traversed forward once, meaning that node j is visited before i if $(j, i) \in A$. The lower bound of a_i^r and the upper bound of b_i^r ($\underline{a}_i^r$ and $\overline{b}_i^r$, respectively) are computed for each node $i \in V$ and resource $r \in \{1, ..., N\}$, using Equations (5.5) and (5.6).

$$\underline{a}_i^r = max\left(l_i^r, \min_{j\in\{\gamma\in V|(\gamma,i)\in A\}}\left(\underline{a}_j^r + b_{j,i}^r\right)\right) \tag{5.5}$$

$$\bar{b}_i^r = min\left(u_i^r, \max_{j\in\{\gamma\in V|(\gamma,i)\in A\}}\left(\bar{b}_j^r + b_{j,i}^r\right)\right) \tag{5.6}$$

Thereafter, the graph is traversed backwards once, meaning that node j is visited before i if $(i,j) \in A$. The feasible bounds at node $i \in V$ are calculated using Equations (5.7) and (5.8).

$$a_i^r = max\left(\underline{a}_i^r, \min_{j\in\{\gamma\in V|(i,\gamma)\in A\}}\left(a_j^r - b_{i,j}^r\right)\right) \tag{5.7}$$

$$b_i^r = min\left(\bar{b}_i^r, \max_{j\in\{\gamma\in V|(i,\gamma)\in A\}}\left(b_j^r - b_{i,j}^r\right)\right) \tag{5.8}$$

Remark that we do not require that $a_i^r \leq b_i^r$. However, node i can not be included in a feasible path if $a_i^r > b_i^r$. This is an *if*-statement, not an *if-and-only-if*-statement. The feasible bounds on each node in Figure 4 are given in Table 1.

Table 1: Resource bounds on the nodes in Figure 4

Node	Bounds $[l_i, u_i]$	Feasible bounds	Dominance bounds
source	$[-\infty, +\infty]$	[0.0, 0.0]	$[-\infty, +\infty]$
n_1	$[-\infty, +\infty]$	[0.0, 0.0]	[0.0, 1.8]
n_2	$[-\infty, +\infty]$	[0.0, 0.0]	No dominance
n_3	$[-\infty, +\infty]$	[0.0, 0.0]	No dominance
n_4	$[-\infty, +\infty]$	[2.4, 3.6]	[2.4, 4.4]
n_5	$[-\infty, +\infty]$	[4.0, 5.8]	No dominance
n_6	$[-\infty, +\infty]$	[8.0, 10.0]	[8.0, 10.0]
n_7	$[-\infty, +\infty]$	[8.0, 8.0]	[8.0, 10.0]
target	[8.0, 10.0]	[8.0, 10.0]	[8.0, 10.0]

The path corresponding with $B_i \in \mathbb{R}^N$ is pruned if $B_i \notin [a_i, b_i]$. Furthermore, a resource vector $B_i \in \mathbb{R}^N$ at node i may only be dominated by another resource vector $B_i' \in \mathbb{R}^N$ if both resource vectors are within the feasible bounds. For the example, the resource consumption on the arc (n_3, n_4) is not allowed in any feasible path as it falls outside the feasible bounds on n_4. The resource value of 2.6 will dominate the resource value of 3.6 in n_4, leading to a feasible path with a resource value of 8.2 in the target. The resource value of 4.0 will dominate the resource value of 5.8 in n_5, not leading to any feasible path. The algorithm produces the path *source* $\rightarrow n_1 \rightarrow n_4 \rightarrow n_6 \rightarrow$ *target*. As we have seen before, this is not the resource feasible path with minimal resource value at the target.

When the heuristic dominance rule is turned on, the resource value 5.8 is dominated by 4.0 in n_5. However, the path from the source to the target via n_5 and n_7 is only feasible when the resource level at n_5 is 5.8. The concept of **strict dominance** bounds is introduced. These are resource bounds on the nodes such that when the resource consumption is within these bounds, then every possible path to the target is feasible. A resource vector $B_i \in \mathbb{R}^N$ at node i may only be dominated by another resource vector $B_i' \in \mathbb{R}^N$ if every feasible path using B_i to the target is also a feasible path when B_i' is used. We will now describe how the dominance bounds are computed in general.

The graph is traversed backwards, meaning that node j is considered before node i if $(i, j) \in A$. For each node, the resource bounds are determined such that every possible path to the target is feasible when the resource consumption is within these bounds. The dominance bounds on node i are given by $[a_i, b_i]$, with $l_i \leq a_i \leq b_i \leq u_i$. It holds that $a_{target} = l_{target}$ and $b_{target} = u_{target}$. Let $a^r_{i,j}$ and $b^r_{i,j}$ be resource bounds on node i for resource $r \in \{1, ..., N\}$, given that arc $(i, j) \in A$ is included in the path. These values are calculated using Equations (5.9) and (5.10).

$$a^r_{i,j} = max\left(\underline{l}^r_i, \left(a^r_j - b^r_{i,j}\right)\right) \tag{5.9}$$

$$b^r_{i,j} = min\left(\bar{u}^r_i, \left(b^r_j - b^r_{i,j}\right)\right) \tag{5.10}$$

The dominance bound on node i for resource r is given by $a^r_{i,j}$ if $a^r_{i,j} = a^r_{i,\gamma}$, for all $j, \gamma \in V$ for which $(i, j) \in A$ and $(i, \gamma) \in A$. Otherwise, no dominance is applied in node i. Moreover, no dominance is applied in node p if no dominance is applied in node i, given that $i, p \in V$ and $(p, i) \in A$.

The dominance bounds on the nodes are given in Table 1. For example, the dominance bounds on n_4 are given by [2.4, 4.4]. The resource consumption on the path from n_4 to the target is 5.6. The resource window associated with the target is [8.0, 10.0]. Therefore, the dominance bounds are given by [8.0 - 5.6, 10.0 - 5.6] = [2.4, 4.4]. No dominance is applied in n_5. The resource value 4.0 does not strictly dominate 5.8 in n_5. The path from the source to the target via n_5 is only feasible when the resource level at n_5 is 5.8. Given that the strict dominance rule is turned on, the solution path is *source* $\rightarrow n_3 \rightarrow n_5 \rightarrow n_7 \rightarrow$ *target* and the resource consumption accumulated along the path from the source to the target is 8.0. The optimal solution is found, but all or almost all paths in the process must be evaluated. This is caused by the fact that some paths are only dominated in the target. The number of paths considered depends on the graph characteristics and the dominance bounds.

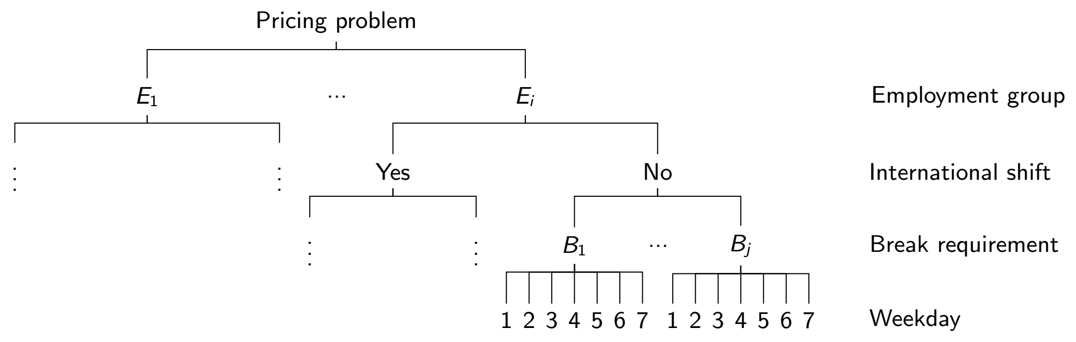
So, all paths are considered when no dominance rule is set and that is potentially very slow. When the heuristic dominance rule is turned on, significantly fewer paths are considered. Unfortunately, it is not guaranteed that the algorithm finds the optimal solution. In terms of column generation, it might occur that the solution process is terminated before the optimal solution of the master problem is found. In that case, at least one resource feasible path with negative reduced cost exists in the graph, but the algorithm is not able to identify this solution variable. The optimal solution variable is found when the strict dominance rule is turned on. However, the algorithm may have to consider all or almost all paths in the graph. Our goal is to design a solution method that solves the CSP to (near-)optimality within a reasonable time limit. Results of computational experiments with each of the dominance rules are described in Chapter 7.

5.3.3.3 Multiple pricing problems

The paths generated inside the pricing problem must obey the naturally occurring constraints. The spread time and required meal break duration depend on the work regime and work function of the employment group that operates the shift, and the shift type. Furthermore, each shift must start and/or end at the home base location of the employment group to which it is allocated. To ensure that the variables satisfy the operational constraints, the pricing problem is split by employment group, nationality and break requirement. We add an additional resource to the labels to offset the night cost and driving time regulation. This resource records the night duration spend, i.e., the shift duration between 10 PM and 6 AM.

Abbink et al. (2011) show that significant performance improvements can be obtained by splitting the pricing problem into multiple, smaller pricing subproblems. Instead of solving the pricing problem for the entire week, they split the pricing problem on weekday. These pricing subproblems must be constructed such that all resource feasible paths in the pricing problem solved for the entire week can still be constructed. We use the following rule to construct the pricing subproblems for each day of the week. Consider a pricing subproblem that corresponds with day i, with $i \in \{1, ..., 7\}$. The maximum spread time of the shifts generated in this subproblem is given by $\bar{d}$. Each crewblock or possible passenger activity that starts after the first time period of day i (12 AM) and ends before the latest time period of the day plus $\bar{d}$ is included in the problem (Bach et al. (2014)). The networks constructed for two successive days overlap in a time length that is equal to the maximum spread time allowed. Figure 5 shows a schematic representation of the pricing subproblems that must be solved.

Figure 5: Rules to split the pricing problem into multiple, smaller pricing subproblems



In general, the pricing problem is split into n pricing subproblems. Let Q be the set $\{1, ..., n\}$. Pricing subproblem $q \in Q$ is stated on the acyclic directed graph $G^q = (V^q, A^q)$. This subproblem asks for the computation of a path from the source $s_q \in V^q$ to the target $t_q \in V^q$ with minimal reduced cost obeying a set of resource constraints.

Recall that solving a pricing problem is very time consuming in general. Significant performance improvements can be obtained if the independent pricing subproblems are solved in parallel. The number of threads depends on the number of CPU cores available. Ideally, there is a thread for each individual pricing subproblem. Not all pricing subproblems need to be solved in each iteration. If the number of threads is smaller than n, the solution time per iteration is reduced by solving only a limited number of pricing subproblems. On the other hand, it is likely that more iterations of the solution process are required before the algorithm terminates.

5.3.3.4 Graph construction and resources

In this paragraph, we describe how the graphs of the pricing subproblems are related to the CSP. First, we give a global description of the graphs. Second, we describe the process to find the set of crewblocks and passenger activities that can be included in the shifts generated inside a specific subnetwork. Third, the node sets and arc sets are described in more detail. Eight resources are introduced to ensure that the generated variables satisfy the naturally occurring constraints.

Global description of the subnetworks

Recall that the pricing problem is split by employment group. As a result, the shifts generated inside a certain pricing problem can be assigned to exactly one employment group. Consider pricing subproblem $q \in Q$. The shifts generated inside this pricing subproblem can be covered by drivers of employment group d_q, with $d_q \in D$. The crewblocks that can be included in the shifts generated inside pricing subproblem q fit in the time interval of the subproblem, and drivers of employment group d_q have the route knowledge, traction knowledge and skill to cover these crewblocks. The passenger activities planned in the shifts generated inside pricing subproblem q also fit in the time interval of the subproblem. Let C_q (K_q) be the set of crewblocks (possible passenger activities) that are relevant in subproblem q. The crewblocks in C_q can not consist of crew requirements that start or end abroad for employment group d_q if national shifts are constructed inside subproblem q.

The source and target of G^q are given by s_q and t_q, respectively. Two nodes are constructed such that we can offset the night cost and driving time regulations. Let $L_q \subseteq L_{d_q}$ be the set of crew depot locations at which the log in and log out activities can be planned. Two nodes are constructed for each location $l \in L_q$. Crewblocks and passenger activities are *events*. Let $\tau_{l,q}$ be the set of unique time points at which an event start or ends at location $l \in L$. Nodes are constructed for each combination of location $l \in L$ and time point $\tau \in \tau_{l,q}$.

An event is represented by an arc between the relevant nodes. Crew deadheading operations, idle time and meal breaks can also be planned on an arc and one arc can have multiple functions. For example, a meal break can be planned between the activities in one crewblock. Eight resources are added to the labels to ensure that the variables obey the naturally occurring constraints.

The set of complex break requirements makes the problem more difficult to solve. The first break must start within some predefined time interval after the start of the shift and it is possible to plan a break on a mobilized locomotive. We can only assure that the constructed variables satisfy the break rules if multiple nodes are constructed for each combination of location $l \in L$ and time point $\tau \in \tau_{l,q}$. The resource windows that apply to node $v \in V^q$ depend on the function of the arc(s) in A^q with a tail node in v. Consider the following example.

Example: Pricing subproblem q is the problem to find a minimum cost path in G^q from the source to the target that is feasible with respect to the resources. In this example, a path is resource feasible if the duration of the path does not exceed $\bar{d}_q$ and if a meal break of at least 30 minutes is planned within the shift. The break must start in the interval [3, 5] hours after the start of the shift. Moreover, crew deadheading may count as a break and the break may be planned on a mobilized locomotive. Crewblock $c_1 \in C_q$ is scheduled on Sunday between 9 AM and 12 PM. No train activities are planned within this crewblock between 10 AM and 11 AM. Nodes n_1 and n_2 correspond with the start and end of crewblock c_1, respectively.

Suppose that the second resource keeps track of the duration of the path. The resource window $[l^2_{n_1}, u^2_{n_1}]$ that applies to n_1 depends on whether or not a break is planned on (n_1, n_2), and on whether or not crewblock c_1 is covered by the path if (n_1, n_2) is included. A crewblock is covered in a shift if the driver performs the actual crew requirements instead of being a passenger on the train.

Let us consider the case in which no break is planned on (n_1, n_2). This arc can be included anywhere in the shift. The duration of (n_1, n_2) is given by ρ_{n_1,n_2}. The resource window $[l^2_{n_1}, u^2_{n_1}]$ is $[0, \bar{d}_q - \rho_{n_1,n_2}]$. Let us consider the case in which the break is planned on (n_1, n_2) and c_1 is not planned on the arc. The resource window $[l^2_{n_1}, u^2_{n_1}]$ is given by $[0.5, min(5, \bar{d}_q - \rho_{n_1,n_2})]$, i.e., if the resource value at node n_1 is 0.5, the meal break can start at 11:30 AM. The break starts at 9 AM if the resource value at n_1 is 5. If a break is planned on (n_1, n_2) and c_1 is associated with the arc, the resource window $[l^2_{n_1}, u^2_{n_1}]$ is given by $[1.5, min(4, \bar{d}_q - \rho_{n_1,n_2})]$. The break must start between 10 AM and 10:30 AM.

Relevant crewblocks and possible passenger activities

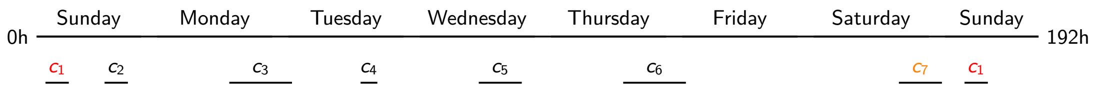
Consider pricing subproblem $q \in Q$. Let $\bar{d}_q$ be the maximum spread time of shifts constructed inside the considered pricing subproblem. $C_{d_q} \subseteq C$ is the set of crewblocks that can be performed by drivers of employment group d_q, i.e., the drivers of employment group d_q have the route knowledge, traction knowledge and skills for all crewblocks in C_{d_q}. Each driving operation can be used to reposition crew by train. The set of crewblocks (possible passenger activities) that can be included in the shifts generated inside G^q is given by C_q (K_q). The crewblocks in C_q can not contain crew requirements that start or end abroad for employment group d_q if national shifts are generated inside pricing subproblem q. Each pricing subproblem must be stated on an acyclic graph, since it is not possible to use a shortest path algorithm on a cyclic network. The crewblocks and passenger activities that are included in the shifts generated inside q fall in the time frame of the pricing subproblem. Consider the following two situations.

- Suppose that the pricing subproblems are split on weekday. In that case, the subnetwork of q is acyclic and it is associated to a particular weekday $i \in \{1, ..., 7\}$. The same crew requirements must be assigned at day i and day $i + 7$. Let $C_i \subseteq C$ $(K_i \subseteq K)$ be the set of crewblocks (possible passenger activities) that start at day i, or at day $i + 1$ not later than $\bar{d}_q$ hours after midnight. C_q (K_q) is the set of crewblocks in $C_i \cap C_{d_q}$ (possible passenger activities in K_i) that end at day i, or at day $i + 1$ not later than $\bar{d}_q$ hours after midnight.
- Suppose that the pricing subproblems are solved over the entire network. This poses a problem to the shortest path algorithm, as the resulting network is cyclic. The problem is solved if we give up that fact that day i is 'equal' to day $i + 7$. C_q (K_q) is the set of crewblocks in C_{d_q} (possible passenger activities) that start later than the first time-point of day 1 and ends not

later than $\bar{d}_q$ hours after midnight at day 8. All feasible shifts that could be constructed in the cyclic network can still be constructed. Consider the following example

> **Example:** The train activities planned in the model week are merged into seven crewblocks, see Figure 6. Crewblock c_1 starts and ends on Sunday ($i = 1$) not later than $\bar{d}_q$ hours after midnight. The crewblock is duplicated, and the start and end time of this duplicate are raised by the duration of one week. Of course, the crewblock must still be assigned only once. Crewblock c_7 starts on Saturday, and ends on Sunday not later than $\bar{d}_q$ hours after midnight. The end time of the crewblock is raised by the duration of one week. The resulting network is acyclic.

Figure 6: Example: Pricing problem solved for the entire week.



Node set

Consider pricing subproblem $q \in Q$. The source and target of G^q are given by s_q and t_q, respectively. Nodes t_q^{day} and t_q^{night} are used to offset the night cost and driving time regulations. Each path constructed inside subproblem q includes one of these nodes. A day shift is constructed if t_q^{day} is included in the path, and a night shift is constructed if t_q^{night} is included. The home base location of employment group d_q is given by l_{d_q}. Let L_q be the set of crew depots at which the log in and log out activities can be planned, i.e. $L_q = \{l_{d_q}\} \cup A_{d_q}$ if $f_{d_q} > 0$ and $L_q = \{l_{d_q}\}$ otherwise. All crew depots located abroad are removed from the set L_q if national shifts are constructed inside subproblem q. Nodes $s_{l,q}$ and $t_{l,q}$ are constructed for every location $l \in L_q$ such that we can offset the overnight cost. The shift starts from (ends up at) crew depot $l \in L_q$ if $s_{l,q}$ ($t_{l,q}$) is included in the path. Let $V^{q,s} = \{s_q, \cup_{l \in L_q} s_{l,q}\}$ and $V^{q,t} = \{t_q, t_q^{day}, t_q^{night}, \cup_{l \in L_q} t_{l,q}\}$.

The required break duration in shifts generated inside subproblem q is given by d_q^{break}. The break can be split into smaller meal breaks with a duration of at least $\underline{d}_q^{break}$ hours. The first break must start in the interval $[\underline{t}_q^{break}, \bar{t}_q^{break}]$ after the start of the shift. Recall that the maximum spread time for shifts constructed inside subproblem q is given by $\bar{d}_q$. The length of these shifts must be greater than or equal to $\underline{d}_q$. The minimum and maximum spread time depend on the work regime of employment group d_q, the relevant break requirements and the nationality of the shift. The possible break duration in crewblock $c \in C_q$ is given by d_c^{break}. The first break in c starts $t_{c,1}$ hours after the start of the crewblock. The latest time point in c such that the remaining break duration is at least $\min\left(d_c^{break}, d_q^{break}\right)$ hours is given by $t_{c,2}$.

Recall that $\tau_{l,q}$ is the set of unique time point at which an event start or ends at location $l \in L$. The start and end time of each possible passenger activity are derived from the vehicle schedule. A directed arc can only exists if the time point associated with the tail node does not exceed the time point associated with the head node. Two crewblocks that are planned in sequence on the same locline cycle can be included in the same shift. We assume that a driver can change from locomotive in 1 minute, i.e., a driver can perform two hand-over activities planned at the same location and time simultaneously. To implement this, the start time and end time of the crewblocks are slightly modified. The start time of crewblock $c \in C_q$ is equal to the start time of $r_{1,c}$ if no hand-over activity is planned before c. Otherwise, the start time of the crewblock is equal to the start time of the hand-over activity planned before c enhanced with 30 seconds (the hand-over duration divided by two). The end time of crewblock $c \in C_q$ is equal to the end time of $r_{n_c,c}$ if no hand-over activity is planned after c. Otherwise, the end time of the crewblock is equal to the start time of the hand-over activity planned after c enhanced with 30 seconds (the hand-over duration divided by two). In this way, the resulting subnetwork is acyclic and successive crewblocks can be combined into one shift.

In the begin of this paragraph, we have described that multiple nodes must be constructed for each combination of location $l \in L$ and time point $\tau \in \tau_{l,q}$ to ensure that the variables constructed inside q satisfy the break requirement. We need to consider the first break in a shift differently from all the others. The first break in subproblem q must start in the interval $[\underline{t}_q^{break}, \bar{t}_q^{break}]$ hours after the start of the shift, while the other breaks must start in the interval $(\underline{t}_q^{break}, \bar{d}_q]$ hours after the start of the shift. Further, the minimum and maximum time between two meal breaks is not specified.

Six node sets are presented in the list below. Node set V^q is expanded with the sets $V^{q,1}$ and $V^{q,2}$, irrespective of the break rules. Further, $V^{q,3}$ ($V^{q,5}$) is added to V^q if $d_q^{break} > 0$ (and if a break can be planned on a mobilized locomotive). V^q is expanded with the set $V^{q,4}$ ($V^{q,6}$) if $d_q^{break} > 0$ and $\underline{d}_q^{break} < d_q^{break}$ (and if a break can be planned on a mobilized locomotive).

$V^{q,1}$ One **general node** is constructed for each combination of location l and time-point $\tau \in \tau_{l,q}$, with $l \in L$ and $\tau \in \tau_{l,q}$. A directed arc is constructed between $v \in V^{q,1}$ and all other nodes in V^q that correspond with same location $l \in L$ and time point $\tau \in \tau_{l,q}$ as node v.

$V^{q,2}$ One **no-break node** is constructed for each combination of location $l \in L$ and time-point $\tau \in \tau_{l,g}$. It is not allowed to plan a break on an arc that has a tail node in $V^{q,2}$. For example, suppose that $v \in V^{q,2}$ corresponds with the start of crewblock $c \in C_q$. If this node is included in the constructed path and if c is covered by the shift, no break is planned in crewblock c.

$V^{q,3}$ One **first-break node** is constructed for each combination of location $l \in L$ and time-point $\tau \in \tau_{l,q}$. A break is planned on each arc that has a tail node in $V^{q,3}$.

$V^{q,4}$ One **break node** is constructed for each combination of location $l \in L$ and time-point $\tau \in \tau_{l,q}$. A break is planned on each arc that has a tail node in $V^{q,4}$.

$V^{q,5}$ One **first-break crewblock node** is constructed for each combination of location $l \in L$, time-point $\tau \in \tau_{l,q}$ and crewblock $c \in C_q$ that starts from l at time τ if a break can be planned in crewblock c. It is possible to plan one or more meal breaks in c if the idle time of the individual breaks is at least equal to the minimum break duration per split, and if it is allowed to plan a break on a mobilized locomotive. A break is planned on each arc that has a tail node in $V^{q,5}$.

$V^{q,6}$ One **break crewblock node** is constructed for each combination of location $l \in L$, time-point $\tau \in \tau_{l,q}$ and crewblock $c \in C_q$ that starts from l at time τ if a break can be planned in crewblock c. A break is planned on each arc that has a tail node in $V^{q,6}$.

Node $v^{q,p}_{l,\tau} \in V^{q,p}$ is associated with location $l \in L$ and time point $\tau \in \tau_{l,q}$, with $p \in \{1, ..., 4\}$. Node $v^{q,p}_{c} \in V^{q,p}$ reflects the start of crewblock $c \in C_q$, with $p \in \{5, 6\}$.

We will now explain how a path generated inside pricing subproblem q looks like. Each path generated inside pricing subproblem q starts with two source nodes. The first node in each path is the source s_q. The second node in the path is an element from the set $V^{q,s} \setminus \{s_q\}$. This node reflects the crew depot at which the log in activity is planned. The path ends with three target nodes. The first target node is $t_{l,q}$, with $l \in L_q$. This node reflects the crew depot at which the log out activity is planned. The second target node is an element from the set $\{t^{day}_q, t^{night}_q\}$, specifying whether a day or night shift is constructed. The last node in the path is the target t_q. In between, we have pairs of nodes, for which one is in $V^{q,1}$ and the other is in $\cup_{p=2}^{6} V^{q,p}$. An arc with a tail node in $V^{q,1}$ connects that node with the nodes in $\cup_{p=2}^{6} V^{q,p}$ that are associated with the same location and time point. The crewblocks, crew repositioning operations, breaks and wait moments are planned on the arcs with a tail node in $\cup_{p=2}^{6} V^{q,p}$ and a head node in $V^{q,1}$. Later in this paragraph, the different arc sets are described.

Resources

Eight resources are added to the labels to ensure that the variables generated inside the individual pricing subproblems obey the naturally occurring constraints. These resources are described in the list below. The resource window $[l^r_i, u^r_i]$ that applies to node $i \in V^q$ is given for each resource $r \in \{1, ..., 8\}$, where $l^r_i, u^r_i \in \mathbb{R}$ and $l^r_i \leq u^r_i$.

1. **Shadow price** The objective of subproblem q is to find the path from s_q to t_q with minimal reduced cost that is feasible with respect to the resources. The resource consumption along the path from the source to the target is the reduced cost of the variable. The resource window associated with node $i \in V^q$ is given by:

$$[l_i^1, u_i^1] = \begin{cases} (-\infty, 0] & \text{if } i = t_q \\ (-\infty, \infty) & \text{otherwise} \end{cases}$$

2. **Spread time** The spread time of each shift constructed in subproblem q must be between $\underline{d}_q$ and $\bar{d}_q$. The resource consumption along the path from the source node to the target node is the duration of the shift, in hours. The resource windows are set such that the first break in the shift starts in the interval $[\underline{t}_q^{break}, \bar{t}_q^{break}]$. The first break is planned on an arc with a tail node in $V^{q,3}$ or $V^{q,5}$. Of course, the other breaks can not be planned within the first $\underline{t}_q^{break} + \underline{d}_q^{break}$ hours after the start of the shift. There is no minimum or maximum time period between two meal breaks. The resource window associated with node $i \in V^q$ is given by:

$$[l_i^2, u_i^2] = \begin{cases} [\underline{t}_q^{break}, \bar{t}_q^{break}] & \text{if } i \in V^{q,3} \\ [\underline{t}_q^{break} + \underline{d}_q^{break}, \bar{d}_q] & \text{if } i \in V^{q,4} \\ [\underline{t}_q^{break} - t_{c,2}, min(\bar{t}_q^{break} - t_{c,1}, \bar{d}_q - d_c)] & \text{if } i = v_c^{q,5} \in V^{q,5} \\ [\underline{t}_q^{break} + \underline{d}_q^{break} - t_{c,2}, \bar{d}_q - d_c] & \text{if } i = v_c^{q,6} \in V^{q,6} \\ [\underline{d}_q, \bar{d}_q] & \text{if } i \in V^{q,t} \\ [0, \bar{d}_q] & \text{otherwise} \end{cases}$$

3. **First break** The first break in a shift is planned on an arc with a tail node in $V^{q,3}$ or $V^{q,5}$. The resource consumption associated with these arcs is equal to 1. The resource consumption along the path from the source s_q to the target t_q must be equal to 1 if a break is required in the shifts constructed inside subnetwork q. This resource is redundant if $d_q^{break} = 0$. Otherwise, the resource window associated with node $i \in V^q$ is given by:

$$[l_i^3, u_i^3] = \begin{cases} [0,0] & \text{if } i \in V^{q,3} \cup V^{q,5} \\ [1,1] & \text{if } i \in V^{q,4} \cup V^{q,6} \cup V^{q,t} \\ [0,1] & \text{otherwise} \end{cases}$$

4. **Break duration** The total duration of the meal breaks in the shift must be larger than or equal to the required break duration that is specified by the break requirements, d_q^{break}. The resource consumption along the path from the source to the target is the total break duration in the shift. We require the resource value to be zero at all nodes $i \in V^{q,3} \cup V^{q,5}$, as no valid break can be scheduled before the first break in the shift. This resource is redundant if $d_q^{break} = 0$. Otherwise, the resource window associated with node $i \in V^q$ is given by:

$$[l_i^4, u_i^4] = \begin{cases} [0,0] & \text{if } i \in V^{q,3} \cup V^{q,5} \\ [d_q^{break}, \bar{d}_q] & \text{if } i \in V^{q,4} \cup V^{q,6} \\ [d_q^{break}, \bar{d}_q] & \text{if } i \in V^{q,t} \\ [0, \bar{d}_q] & \text{otherwise} \end{cases}$$

5. **Driving time** The total driving time of all crew requirements included in the path may not exceed the maximum driving time, which depends on whether or not the constructed variable is a night shift. The resource consumption along the path from the source to the target is the total duration of all driving operations included in the shift. Let $\bar{drive}_q^{day}$ and $\bar{drive}_q^{night}$ be the maximum driving time allowed in day shifts and night shifts, respectively. The resource window associated with node $i \in V^q$ is given by:

$$[l_i^5, u_i^5] = \begin{cases} [0, \bar{drive}_q^{day}] & \text{if } i = t_q^{day} \\ [0, \bar{drive}_q^{night}] & \text{if } i = t_q^{night} \\ [0, \max(\bar{drive}_q^{day}, \bar{drive}_q^{night})] & \text{otherwise} \end{cases}$$

6. **Night duration** There are different costs and driving time regulations depending on whether or not the shift is a night shift. This resource records the night duration spend. The resource consumption along the path from the source to the target is the total spread time of the shift between 10 PM and 6 AM. The resource value at the target node must be smaller than 3 if a day shift is constructed, and it

must be greater than or equal to 3 if a night shift is constructed. The resource window associated with node $i \in V^q$ is given by:

$$[l_i^6, u_i^6] = \begin{cases} [0, 3) & \text{if } i = t_q^{day} \\ [3, \bar{d}_q] & \text{if } i = t_q^{night} \\ [0, \bar{d}_q] & \text{otherwise} \end{cases}$$

7. **International** The operational constraints depend on whether the constructed shift is an international shift for employment group d_q. This resource records the number of international activities covered by the shift, and the number of crew depots visited that are located abroad. A positive resource consumption accumulated along the path from the source to the target indicates that an international shift is constructed. This resource is redundant if national shifts are constructed inside subproblem q, or if the same restrictions hold for national and international shifts. Otherwise, the resource window associated with node $i \in V^q$ is given by:

$$[l_i^7, u_i^7] = \begin{cases} [1, \infty) & \text{if } i = t_q \\ [0, \infty) & \text{otherwise} \end{cases}$$

8. **Home** Drivers connected to employment group $d \in D$ may rest at a hotel away from their home base for at most one consecutive night if $f_d > 0$. The resource consumption along the path from the source node to the target node is equal to one if the shift starts or ends at an away location of employment group d_q, and two if the shift starts and ends at the home base location of the group. The resource window associated with node $i \in V^q$ is given by:

$$[l_i^8, u_i^8] = \begin{cases} [1, 2] & \text{if } f_{d_q} > 0 \\ [2, 2] & \text{otherwise} \end{cases}$$

Arc set

Consider pricing subproblem $q \in Q$. The following parameters are introduced to specify the resource consumptions associated with the arc $(i,j) \in A^q$. We know that the time point associated with node i does not exceed the time point associated with node j if $(i,j) \in A^q$, for $i,j \in V^q$.

$\rho_{i,j}$	The duration of arc (i,j).
$\rho_{i,j}^{claim}$	The total duration of arc (i,j) on Monday before 4 AM and on Friday after 7 PM.
$\rho_{i,j}^{night}$	The night duration spend on arc (i,j).
$\rho_{i,j}^{npro}$	The non-productive time on arc (i,j).
$\rho_{i,j}^{break}$	The possible break duration on arc (i,j). Time windows are not considered yet.
$\rho_{i,j}^{HLP}$	The total duration of the crew deadheading activities planned on arc (i,j). An arc is contructed for every driving operation in K_q and $\rho_{i,j}^{HLP}$ is equal to the duration of this operation. As mentioned before, several other vehicle types can be used to transfer crew from one location to another. In that case, $\rho_{i,j}^{HLP}$ is equal to the duration of the fastest option to transfer crew from the location of node i to the location of node j within two deadheading operations. If crew repositioning is not possible between this pair of locations within two deadheading operations, an expensive taxicab can be used. For simplicity, we assume that the time needed to transfer crew with the expensive taxicab only depends on whether or not the locations are in the same country.
$c_{i,j}^{HLP}$	The cost of the crew deadheading activities planned on arc (i,j).

In the remainder of this paragraph, we present the subsets of arcs constructed for pricing subproblem q. We present them in the order from the easier ones (that correspond to crewblocks that we need assign or deadheading operations) to the more difficult ones (arcs included to model certain costs or constraints). The resource consumption is equal to zero if it is not explicitly specified.

For now, we assume that $d_q^{break} > 0$ and $\underline{d}_q^{break} < d_q^{break}$. In practice, not all subsets of nodes described on page 46 and 47 are added to V^q if $d_q^{break} = 0$ or $\underline{d}_q^{break} = d_q^{break}$. In that case, not all arcs described below are added to A^q. More specific, the arcs with a node in $\cup_{p=2}^{6} V^{q,p}$ are not constructed if $d_q^{break} = 0$. If $\underline{d}_q^{break} = d_q^{break}$ and $d_q^{break} > 0$, the arcs with a node in $V^{q,4} \cup V^{q,6}$ are not constructed.

Crewblock Consider crewblock $c \in C_q$. The crewblock is represented by the arc between $i \in V^{q,2}$ corresponding with the start of crewblock c and $j \in V^{q,1}$ corresponding with the end of the crewblock. Furthermore, an arc representing crewblock c is constructed between each node $i \in \{v_c^{q,5}, v_c^{q,6}\}$ and j if a break can be planned on crewblock c. The break duration planned on this arc is equal to the minimum value of d_c^{break} and d_q^{break}. The driver performs the crew requirements in crewblock c if the arc (i,j) is included in the path.

- $b_{i,j}^1 = \beta_c - \delta_c + \rho_{i,j}(\lambda_{d_g} - \sigma_{d_q}) + c^{npro}\rho_{i,j}^{npro} + c^{claim}\rho_{i,j}^{claim} + \zeta_d \rho_{i,j}$
- $b_{i,j}^2 = \rho_{i,j}$
- $b_{i,j}^3 = 1 \quad \text{if } i = v_c^{q,5}$
- $b_{i,j}^4 = \rho_{i,j}^{break}$ if $i \in \{v_c^{q,5}, v_c^{q,6}\}$
- $b_{i,j}^5 = \rho_c^{drive}$
- $b_{i,j}^6 = \rho_{i,j}^{night}$
- $b_{i,j}^7 = 1$ if crewblock c contains crew requirements that start or end at a location abroad for employment group d_q.

Passenger activity The passenger activities that can be included in the shift generated inside subproblem q are the elements of the set K_q. Consider driving operation $k \in K_q$. This activity starts from location l_1 at time point τ_1, and ends up at location l_2 at time point τ_2. An arc is constructed between $i = v_{l_1,\tau_1}^{q,2}$ and $j = v_{l_2,\tau_2}^{q,1}$, representing a passenger activity on k without a break. Furthermore, an arc is constructed between each node $i \in \{v_{l_1,\tau_1}^{q,3}, v_{l_1,\tau_1}^{q,4}\}$ and j if crew deadheading may count as a break and $\tau_2 - \tau_1 \geq \underline{d}_q^{break}$. The break duration on the arc (i,j) is equal to $\tau_2 - \tau_1$.

- $b_{i,j}^1 = \rho_{i,j}(\lambda_{d_g} - \sigma_{d_g}) + c^{npro}\rho_{i,j}^{npro} + c_{i,j}^{HLP} + c^{claim}\rho_{i,j}^{claim} + \zeta_d \rho_{i,j}$
- $b_{i,j}^2 = \rho_{i,j}$
- $b_{i,j}^3 = 1 \quad \text{if } i \in V^{q,3}$
- $b_{i,j}^4 = \rho_{i,j}^{break}$ if $i \in V^{q,3} \cup V^{q,4}$
- $b_{i,j}^6 = \rho_{i,j}^{night}$

Crew deadheading Foot, lease car, own car and taxi can be used for crew deadheading operations others than as a passenger on a train. The deadhead duration between the locations l_1 and l_2 is given by d_{l_1,l_2}^{HLP}, for $l_1, l_2 \in L$ and $l_1 \neq l_2$. An expensive taxicab is available if it is not possible to transfer crew from l_1 to l_2 within two deadheading operations. None of these vehicle types has a fixed time schedule, and it is theoretically possible to have a crew deadheading arc between two locations at any point in time. This case is not manageable, since the number of crew deadheading arcs would be extremely large. The number of deadheading possibilities should be reduced by omitting many possible crew deadheading arcs, without reducing the solution space too much.

Consider the nodes $v = v^{q,2}_{l,\tau}$ and $w = v^{q,1}_{l_1,\tau_1}$, with $l, l_1 \in L$, $l \neq l_1$, $\tau \in \tau_{l,q}$ and $\tau_1 \in \tau_{l_1,q}$. Many possible crew deadheading arcs are omitted if the arc (v, w) is only constructed if $d^{HLP}_{l,l_1} \leq \tau_1 - \tau$ and if there is no other node $i \in V^{q,1}$ associated with location l_1 for which $d^{HLP}_{l,l_1} \leq \rho_{v,i} < \tau_1 - \tau$. Consider the nodes $v \in \{v^{q,3}_{l,\tau}, v^{q,4}_{l,\tau}\}$ and $w = v^{q,1}_{l_1,\tau_1}$, with $l, l_1 \in L$, $l \neq l_1$, $\tau \in \tau_{l,q}$ and $\tau_1 \in \tau_{l_1,q}$. The arc (v, w) is only constructed if $d^{HLP}_{l,l_1} \leq \tau_1 - \tau$ and $\rho^{break}_{v,w} \geq \underline{d}^{break}_q$, and if there is no other node $i \in V^{q,1}$ associated with location l_1 for which $d^{HLP}_{l,l_1} \leq \rho_{v,i} < \tau_1 - \tau$ and $\rho^{break}_{i,w} \geq \underline{d}^{break}_q$. The break on arc (v, w) is the sum of (1) d^{HLP}_{l,l_1} if $d^{HLP}_{l,l_1} \geq \underline{d}^{break}_q$ and if a break can be planned during repositioning, and (2) $\tau_1 - \tau - d^{HLP}_{l,l_1}$ if $\tau_1 - \tau - d^{HLP}_{l,l_1} \geq \underline{d}^{break}_q$ and if there is a break room present at l or l_1.

- $b^1_{v,w} = \rho_{v,w}(\lambda_{d_q} - \sigma_{d_q}) + c^{npro}\rho^{npro}_{v,w} + c^{HLP}_{v,w} + c^{claim}\rho^{claim}_{v,w} + \zeta_d \rho_{v,w}$.
- $b^2_{v,w} = \rho_{v,w}$
- $b^3_{v,w} = 1$ if $v \in V^{q,3}$
- $b^4_{v,w} = \rho^{break}_{v,w}$ if $v \in V^{q,3} \cup V^{q,4}$
- $b^6_{v,w} = \rho^{night}_{v,w}$

Wait arc A driver can wait at a location before he continues with another task. It is theoretically possible to have a wait arc at a location at any point in time. Of course, this case is not manageable. The number of wait arcs should be reduced by omitting many possible wait arcs.

Consider the time point $\tau_1 \in \tau_{l,q}$ at which an event takes place at location $l \in L$. Let τ_2 be the smallest element in $\tau_{l,q}$ for which it holds that $\tau_2 > \tau_1$. A wait arc is constructed between $i = v^{q,2}_{l,\tau_1}$ and $j = v^{q,1}_{l,\tau_2}$. Let τ_3 be the smallest element in $\tau_{l,q}$ for which it holds that $\tau_3 \geq \tau_1 + \underline{d}^{break}_q$. Let τ_4 be the smallest element in $\tau_{l,q}$ for which it holds that $\tau_4 \geq \tau_1 + d^{break}_q$. An arc is constructed between $i \in \{v^{q,3}_{l,\tau_1}, v^{q,4}_{l,\tau_1}\}$ and $j = v^{q,1}_{l,\tau}$ if a break room is present at location l, for all $\tau \in \{\chi \in \tau_{l,q} | \tau_3 \leq \chi \leq \tau_4\}$.

- $b^1_{i,j} = \rho_{i,j}(\lambda_{d_q} - \sigma_{d_q}) + c^{npro}\rho^{npro}_{i,j} + c^{claim}\rho^{claim}_{i,j} + \zeta_d \rho_{i,j}$
- $b^2_{i,j} = \rho_{i,j}$
- $b^3_{i,j} = 1 \qquad$ if $i \in V^{q,3}$
- $b^4_{i,j} = \rho^{break}_{i,j}$ if $i \in V^{q,3} \cup V^{q,4}$
- $b^6_{i,j} = \rho^{night}_{i,j}$

General arc An arc is constructed between node $v \in V^{q,1}$ and each node $w \in \cup_{p=2}^{6} V^{q,p}$ that is associated with the same location and time point as node v. Different resource windows are associated with the head nodes of these arcs, such that the complex break requirements can be taken into account.

Departure arc Each shift starts with a log in activity planned at a crew depot $l \in L_q$. Crew deadheading is used to reposition the driver from the crew depot to the start location of the first crewblock covered by the shift.

A departure arc is constructed between node $i = s_{l_1,q}$ and each node $j \in \cup_{\tau \in \tau_{l_2,q}} \{v_{l_2,\tau}^{q,1}\}$ if $d_{l_1,l_2}^{HLP} \leq \bar{d}_q$, for all $l_1 \in L_q$ and $l_2 \in L$. The break duration on (i,j) is equal to zero if crew deadheading can not count as a break. Furthermore, the break duration is equal to zero if the driving time after $\underline{t}_q^{break}$ hours after the start of the shift is smaller than $\underline{d}_q^{break}$, as the first crew repositioning activity is scheduled directly after the log in activity. The start day of the shift is given by $u_{i,j}$ when (i,j) is included in the path, with $1 \leq u_{i,j} \leq 7$.

- $b_{i,j}^1 = \rho_{i,j}(\lambda_d - \kappa_d) - \epsilon_{d,u_{i,j}} - \eta_d + c^{npro}\rho_{i,j}^{npro} + c_{i,j}^{HLP} + c^{claim}\rho_{i,j}^{claim} + \zeta_d(\rho_{i,j} - d_q^{break})$ if i is associated with the home base location of d_q.
- $b_{i,j}^1 = \rho_{i,j}(\lambda_d - \kappa_d) - \epsilon_{d,u_{i,j}} - \eta_d + \alpha_{d_q,l,u_{i,j}}^{-} - \alpha_{d_q,l,u_{i,j}}^{+} + c^{npro}\rho_{i,j}^{npro} + c_{i,j}^{HLP} + c^{claim}\rho_{i,j}^{claim} + \zeta_d(\rho_{i,j} - d_q^{break})$ if i is associated with crew depot $l \in L \setminus \{l_{d_q}\}$.
- $b_{i,j}^2 = \rho_{i,j}$
- $b_{i,j}^3 = 1$ if $\rho_{i,j}^{break} > 0$
- $b_{i,j}^4 = \rho_{i,j}^{break}$
- $b_{i,j}^6 = \rho_{i,j}^{night}$

Recall that the first break in each shift generated inside subproblem q must start within the interval $[\underline{t}_q^{break}, \bar{t}_q^{break}]$ after the start of the shift. Initially, we did not include an additional departure arc if no break can be planned on (i,j), with j corresponding to the start of a crewblock. Unfortunately, we discovered that some crewblocks could not be included in a feasible shift, due to break violations. It was not possible to plan a break before, during or after the crewblock that addresses the break requirements. We decided to construct an additional departure arc between nodes i and j if the following four constraints are fulfilled.

1. No break can be planned on the current arc (i,j), i.e., $\rho_{i,j}^{break} = 0$,

2. At least one crewblock $c \in C_q$ starts at the time point and location associated with node j,
3. Crew deadheading may count as a break and $\rho_{i,j}^{HLP} \geq \underline{d}_q^{break}$, or there is a break room present at location l_1 or location l_2,
4. The required break duration is larger than zero, i.e., $d_q^{break} > 0$.

Let $(i,j)'$ be the arc constructed between node i and node j if the four constraints listed above are satisfied. The meal break planned on this arc must have a duration in the interval $[\underline{d}_q^{break}, d_q^{break}]$. We chose to construct the arcs such that the total break duration on the arc is minimized, but at least $\underline{d}_q^{break}$. In that case, the arc duration is minimized. Another option is to construct more than two departure arcs between node i and node j, each with another break duration and thus another arc length. Let d^{login} be the duration of the log in activity. The meal break is planned according to the following rules.

1. If crew deadheading may not count as a break or if $\rho_{i,j}^{HLP} < \underline{d}_q^{break}$, the meal break must be planned at location l_1 or location l_2. To minimize the impact of traffic jams on the vehicle schedule and to minimize the , the meal break is planned at location l_2 if there is a canteen available at that location. The break starts x hours after the start of the shift, with $x = \max(d^{login} + \rho_{i,j}^{HLP}, \underline{t}_q^{break})$. The first crew deadheading activity is planned directly after the log in activity. The meal break is planned at location l_1 if there is a break room present at this location and no canteen is available at location l_2. The break starts $\underline{t}_q^{break}$ hours after the start of the shift. The first crew deadheading activity is planned directly after the break,
2. If crew deadheading may count as a break and $\rho_{i,j}^{HLP} \geq \underline{d}_q^{break}$, the break is planned during crew repositioning and starts $\underline{t}_q^{break}$ hours after the start of the shift. The remaining crew deadheading duration is $\underline{d}_q^{break}$. The crew repositioning operation starts $\underline{t}_q^{break} - (\rho_{i,j}^{HLP} - \underline{d}_q^{break})$ hours after the start of the shift.

- $b^1_{i,j'} = \rho_{i,j'}(\lambda_d - \kappa_d) - \epsilon_{d,u_{i,j'}} - \eta_d + c^{npro}\rho^{npro}_{i,j'} + c^{HLP}_{i,j'} + c^{claim}\rho^{claim}_{i,j'} + \zeta_d(\rho_{i,j'} - d_q^{break})$ if i is associated with the home base location of d_q.
- $b^1_{i,j'} = \rho_{i,j'}(\lambda_d - \kappa_d) - \epsilon_{d,u_{i,j'}} - \eta_d + \alpha^-_{d_q,l,u_{i,j'}} - \alpha^+_{d_q,l,u_{i,j'}} + c^{npro}\rho^{npro}_{i,j'} + c^{HLP}_{i,j'} + c^{claim}\rho^{claim}_{i,j'} + \zeta_d(\rho_{i,j'} - d_q^{break})$ if i is associated with crew depot $l \in L \setminus \{l_{d_q}\}$.
- $b^2_{i,j'} = \rho_{i,j'}$

- $b^3_{i,j'} = 1$
- $b^4_{i,j'} = \rho^{break}_{i,j'}$
- $b^6_{i,j'} = \rho^{night}_{i,j'}$

Arrival arc Each shift ends with a log out activity planned at a crew depot $l \in L_q$. Crew deadheading is used to reposition the driver from the end location of the last crew requirement covered by the shift to the crew depot. An arc is constructed between $i \in \cup_{\tau \in \tau_{l_1,q}} \{v^{q,2}_{l_1,\tau}\}$ and $j = t_{l_2,q}$ if $d^{HLP}_{l_1,l_2} \leq \bar{d}^q$, for all $l_1 \in L$ and $l_2 \in L_q$. Furthermore, an arc is constructed between $i \in \cup_{\tau \in \tau_{l_1,q}} \{v^{q,3}_{l_1,\tau}, v^{q,4}_{l_1,\tau}\}$ and $j = t_{l_2,q}$ if $\underline{d}^{break}_q \leq d^{HLP}_{l_1,l_2} \leq \bar{d}^q$ and crew deadheading may count as a break, for all $l_1 \in L$ and $l_2 \in L_q$. Suppose that (i, j) is included in the path. The end day of the resulting shift is computed and given by $z_{i,j}$, with $1 \leq z_{i,j} \leq 7$. The parameter $z^-_{i,j}$ represents the day before $z_{i,j}$, i.e., $z^-_{i,j} = z_{i,j} - 1$ if $z_{i,j} \geq 2$ and $z^-_{i,j} = 7$ if $z_{i,j} = 1$

- $b^1_{i,j} = \rho_{i,j}(\lambda_d - \kappa_d) + c^{npro}\rho^{npro}_{i,j} + c^{HLP}_{i,j} + c^{claim}\rho^{claim}_{i,j} + \zeta_d \rho_{i,j}$ if j is associated with the home base location of d_q.
- $b^1_{i,j} = \rho_{i,j}(\lambda_d - \kappa_d) - \alpha^-_{d_q,l,z^-_{i,j}} + \alpha^-_{d_q,l,z^-_{i,j}} + c^{npro}\rho^{npro}_{i,j} + c^{HLP}_{i,j} + c^{claim}\rho^{claim}_{i,j} + \zeta_d \rho_{i,j}$ if j is associated with crew depot $l \in L \setminus \{l_{d_q}\}$.
- $b^2_{i,j} = \rho_{i,j}$
- $b^3_{i,j} = 1$ if $i \in V^{q,3}_{l_1}$.
- $b^4_{i,j} = \rho^{break}_{i,j}$ if $i \in V^{q,3}_{l_1} \cup V^{q,4}_{l_1}$
- $b^6_{i,j} = \rho^{night}_{i,j}$

Source arc An arc is constructed between the source s_q and node $s_{l,q}$, for all $l \in L_q$. The shift starts from depot location $l \in L_q$ if $(s_q, s_{l,q})$ is included in the path. A schematic representation of the source arcs is shown in Figure 7 and Figure 8.

- $b^7_{s_q,s_{l,q}} = 1$ if $l \in L_q \setminus \{l_{d_q}\}$ and l is located abroad for d_q
- $b^8_{s_q,s_{l,q}} = 1$ if $l = l_{d_q}$

Target arc An arc is constructed between each node $v \in \{t^{day}_q, t^{night}_q\}$ and the target t_q. A day shift is constructed if (t^{day}_q, t_q) is included in the path, and a night shift is constructed if arc (t^{night}_q, t_q) is included in the path. Furthermore, an arc is constructed between $t_{l,q}$ and each node $v \in \{t^{day}_q, t^{night}_q\}$, for all $l \in L_q$. The shift ends at depot location $l \in L_q$ if $(t_{l,q}, t^{day}_q)$ or $(t_{l,q}, t^{night}_q)$ is included in the path. A schematic representation of the target arcs is shown in Figure 7 and Figure 8.

- $b^{1}_{t^{night}_q, t_q} = c^{night}$
- $b^{1}_{t_{l,q}, v} = f_{d_q}\xi_{d_q}$ if $v \in \{t^{day}_q, t^{night}_q\}$ and $l = l_{d_q}$
- $b^{1}_{t_{l,q}, v} = (f_{d_q} - 1)\xi_{d_q} + c^{overnight}$ if $v \in \{t^{day}_q, t^{night}_q\}$ and $l \in L_q \setminus \{l_{d_q}\}$
- $b^{7}_{t_{l,q}, v} = 1$ if $v \in \{t^{day}_q, t^{night}_q\}$ and $l \in L_q \setminus \{l_{d_q}\}$ is located abroad for d_q
- $b^{8}_{t_{l,q}, v} = 1$ if $v \in \{t^{day}_q, t^{night}_q\}$ and $l = l_{d_q}$

Figure 7: Schematic representation of the source arcs and target arcs in the acyclic directed graph G^q given that $f_{d_q} = 0$ and $L_q = \{l_1\}$, with $q \in Q$.

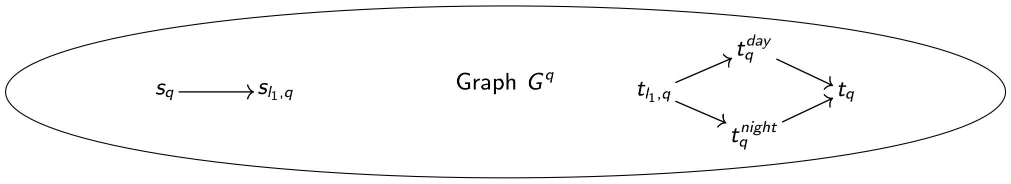
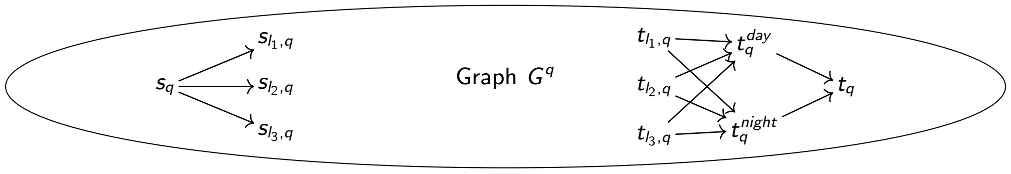


Figure 8: Schematic representation of the source arcs and target arcs in the acyclic directed graph G^q given that $f_{d_q} > 0$ and $L_q = \{l_1, l_2, l_3\}$, with $q \in Q$.



6 Data description

In this chapter, four problem instances are described for which the optimal solution to crew scheduling is known. Furthermore, different problem instances are derived from the real-life application of the European rail-cargo company. These datasets are used to evaluate the robustness of the method to different problem sizes and implementation options. The aim of Section 6.1 is to set the parameter values that are the same in each problem instance. The problem instances for which the optimal solutions are known are described in Section 6.2. Information about the locations and crew deadheading options, loclines and employment groups included in the original problem is given in Section 6.3. Problem instances derived from this real-life application are described in Section 6.4.

6.1 Fixed parameter values

Some parameter values are fixed. These values are used in each problem instance that is considered in this thesis. The aim of this section is to specify these values.

Passenger activities are planned to make the crew schedule more efficient. However, disruption during daily operations have more impact on the crew schedule and vehicle schedule if passenger activities are planned on the disrupted locomotive. Therefore, the cost for driving as a passenger on a train is 2 euro per hour. Besides crew repositioning by train, four other crew deadheading options can be used. The line segments on which each of these crew deadheading options can be used and the duration of the trips are specified by the data. The costs for the different crew deadheading options are depicted in Table 2.

Table 2: Costs for the different crew deadheading options (in euro per hour)

Crew deadheading option	Foot	Lease car	Own car	Taxi	Train
Cost (in euro per hour)	2	20	15	25	2

The salary cost and fixed cost per driver of employment group $d \in D$ are given by the parameters ζ_d and γ_d, respectively. In practice, these costs depend on the work function, work regime, route knowledge, traction knowledge and skills of the employee. In this thesis, we assume that the hourly wage of each driver is 30 euro and that the fixed cost per driver is 1000 euro per week. It is worth mentioning that it is easy to incorporate cost parameters that depend on the employment group of the driver, since each shift is constructed for exactly one employment group.

Recall that the minimum and maximum average weekly workload per driver of employment $d \in D$ are given by the parameters $\underline{V}_d$ and $\bar{V}_d$, respectively. In this thesis, the average weekly workload per crew member must be larger than or equal to 36 hours hours, for all $d \in D$. This constraint is considered to be a soft constraint. Therefore, a penalty cost of $\psi_d = 1$ euro per hour is incurred if the average weekly workload of a crew member of employment group $d \in D$ is below $\underline{V}_d$. Furthermore, the average weekly workload per driver may not exceed 44 hours. This constraint is considered to be a hard constraint, and it should be satisfied at all times.

An overnight stay at a hotel costs 200 euro ($c^{overnight}$). Crew planners strive for more effective use of the available crew to reduce the number of crews needed. Crew requirements, log in activities and log out activities are regarded productive. Non-productive time in a shift is penalized with 5 euro per hour (c^{npro}). Furthermore, c^{claim} is 10 euro per hour, c^{night} is 50 euro per night shift, θ is $10,000$ euro per hour and ν is 30 euro per hour.

The objective function of (4.1) - (4.15) is given by Equation (6.1).

$$\begin{aligned}
z =& \sum_{s \in S} c_s X_s + \sum_{c \in C} (u_c U_c + o_c O_c) + \sum_{d \in D} (\gamma_d Y_d + \psi_d \Psi_d) \qquad (6.1)\\
=& \sum_{s \in S} \left(\sum_{d \in D} \left(\zeta_d a^2_{s,d} (d_s - d^{break}_s) \right) + c^{HLP}_s + c^{claim} d^{claim}_s + c^{npro} d^{npro}_s + c^{night} a^{night}_s \right) X_s \\
&+ \sum_{c \in C} (\theta \mu_c U_c + \nu d_c O_c) + \sum_{d \in D} \left(\gamma_d Y_d + \psi_d max(\underline{V}_d Y_d - \sum_{s \in S} a^2_{s,d} d_s X_s, 0) \right) \\
=& \sum_{s \in S} \left(30(d_s - d^{break}_s) + c^{HLP}_s + 10 d^{claim}_s + 5 d^{npro}_s + 50 a^{night}_s \right) X_s \\
&+ \sum_{c \in C} (10,000 \mu_c U_c + 30 d_c O_c) + \sum_{d \in D} \left(1000 Y_d + max(36 Y_d - \sum_{s \in S} a^2_{s,d} d_s X_s, 0) \right)
\end{aligned}$$

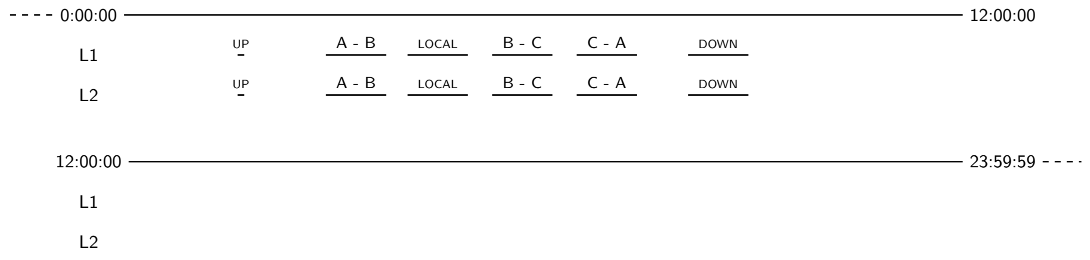
6.2 Problem instances for which the optimal solution is known

In this section, small problem instances are described for which the optimal solution to crew scheduling is known. The number of train activities planned in the model week is between 84 and 120. At most two employment groups have the route knowledge, traction knowledge and skills for all crew requirements. Information about the employment groups, crew requirements, crew repositioning options and relevant operational constraints is given. The optimal solution of each instance is described.

6.2.1 Problem instance one

Train activities, which take place at three relief locations, are planned on two identical loclines. The sequence of train activities is the same at each day of the week. A schematic representation of the vehicle schedule for one day of the week is shown in Figure 9. The locomotives are mobilized between 2 AM and 9 AM.

Figure 9: Schematic representation of the train activities planned per day of the week in problem instance one. Twelve train activities are planned on two locomotives. The train activities are planned at three relief locations, between 2 AM and 9 AM



Two employment groups with global drivers have the traction knowledge, route knowledge and skills for all 84 crew requirements. Hence, $D = \{d_1, d_2\}$ and $a^6_{c,d} = 1$ for all $c \in C$ and $d \in D$. At most ten crew rosters can be allocated to each employment group. A break room is present at the relief location. The home base location of employment group d is given by l_d, for all $d \in D$. Crew deadheading by foot is used to transfer crew from their home base location to the relief locations, and back. Crew repositioning between home base location l_{d_1} (l_{d_2}) and each of the relief locations takes 10 (20) minutes.

The spread time of each shift generated inside this problem must be between 6 and 10 hours. The total duration of all train services and rolling stock deadheading operations included in a shift may not exceed 8 hours. No break is required.

In the **optimal solution**, fourteen shifts are added to the crew schedule. Two shifts are sufficient to cover the crew requirements planned at one day of the week, each covering six crew requirements. Even though the effect of delay is minimized if the number of crew changes on a mobilized locomotive is minimized, no additional cost is incorporated with a crew change. Recall that we have assumed that a driver can perform two hand-over activities at the same time, given that these crew activities are planned at the same relief location. As a result, each shift can contain crew requirements planned on both loclines. Crew repositioning and salary costs are minimized if the shifts start and end at the home base location of the first employment group. Three crew rosters must be constructed to cover all fourteen shifts.

The objective value of the optimal solution is 7100.33 euro. The components of the solution value are specified by Equation (6.2).

$$\begin{aligned} z =& \sum_{s \in S}\Big(30(d_s - d_s^{break}) + c_s^{HLP} + 10d_s^{claim} + 5d_s^{npro} + 50a_s^{night}\Big)X_s \\ &+ \sum_{c \in C}(10,000\mu_c U_c + 30d_c O_c) + \sum_{d \in D}\Big(1000Y_d + max(36Y_d - \sum_{s \in S} a^2_{s,d} d_s, 0)\Big) \\ =& \; 14x30x(7\frac{7}{12} - 0) + 14x2x\frac{1}{3} + 10x2x2\frac{1}{3} + 5x14x2\frac{1}{4} + 50x14 \\ &+ 10,000x0 + 30x0 + 1000x3 + \frac{37}{60}x3 \\ \approx& \; 7100.33 \; euro \end{aligned} \tag{6.2}$$

Each shift in the optimal solution starts at 1:40 AM with a log in activity scheduled at l_{d_1}. The log out activity is scheduled at l_{d_1} as well and ends at 9:15 AM. The paid time per shift is equal to the spread time of 7 hours and 35 minutes, since no break is required. Crew repositioning from and to the depot location takes 20 minutes per shift. Two shifts have a duration of 2 hours and 20 minutes on Monday before 4 AM. The non-productive time per shift is 2 hours and 15 minutes. All selected shifts are night shifts. Three employees are needed to cover the shifts allocated to employment group d_1 and the average weekly workload per employee is 35 hours and 23 minutes.

6.2.2 Problem instance two

Train activities, which take place at two relief locations, are planned on two loclines. The same train activities are planned on Monday till Saturday. No train activities are scheduled on Sunday. A schematic representation of the train activities per day (except Sunday) is shown in Figure 10. The locomotive allocated to L_1 is mobilized between 0:45 AM and 5:45 AM, and between 00:45 PM and 5:45 PM. The locomotive allocated to L_2 is mobilized between 6:15 AM and 11:15 AM, and between 6:15 PM and 11:15 PM.

The global drivers that are part of the employment group in this dataset have the route knowledge, traction knowledge and skills for all 120 crew requirements. The maximum number of crew rosters that can be allocated to the employment group is ten. A break room is present at the relief locations. The shift length must be between 5 and 12 hours. The total duration of all train services and rolling stock deadheading operations included in a shift may not exceed 8 hours. No break is required. Crew repositioning by taxi takes 5 minutes between each pair of locations.

The **optimal solution** is not unique. At least twelve shifts are needed to cover all crew requirements. The objective value of the optimal solution is 7649.17 euro, see Equation (6.3).

$$\begin{aligned} z =& \sum_{s\in S}\Big(30(d_s - d_s^{break}) + c_s^{HLP} + 10d_s^{claim} + 5d_s^{npro} + 50a_s^{night}\Big)X_s \\ &+ \sum_{c\in C}\Big(10,000\mu_c U_c + 30d_c O_c) + \sum_{d\in D}(1000Y_d + max(36Y_d - \sum_{s\in S} a_{s,d}^2 d_s, 0)\Big) \\ =& \ 12x30x(10\frac{11}{12} - 0) + 12x25x\frac{1}{6} + 10(3\frac{1}{2} + 4\frac{5}{12}) + 5x12x4\frac{5}{6} + 50x6 \\ &+ 10,000x0 + 30x0 + 1000x3 + 0 \\ \approx& \ 7649.17 \ euro \end{aligned} \tag{6.3}$$

All crew requirements that start before (after) noon at the same day are included in one shift. Six shifts start at 0:30 AM and end at 11:25 AM. The other shifts start at 00:30 PM and end at 11:25 PM. The paid time is equal to the spread time of the shift, as no break is required. The crew deadheading duration per shift is 10 minutes. The total shift duration on Monday before 4 AM is 3 hours and 30 minutes and the total shift duration on Friday after 7 PM is 4 hours and 25 minutes. The non-productive time per shift is 4 hours and 50 minutes. The shifts that start at 0:30 AM are night shifts. Three employees are needed to cover the selected shifts and the average weekly workload per employee is 43 hours and 40 minutes.

As mentioned before, the optimal solution is not unique. We will show this with an example. The cost of the solution does not change if a shift that covers all crew requirements planned before (after) noon at the same day is split into two shifts, each covering the sequence of crew requirements planned on one locline. The idle time between the shut-down activity planned on L_1 and the start up activity planned on L_2 is 30 minutes. The increase in crew deadheading cost (25 x 1 / 6) is the same as the decrease in non-productive cost and salary cost (5 x 1 / 3 + 30 x 5 / 60). The maximum number of shifts selected in an optimal solution is fifteen, i.e., at most three shifts are split. Otherwise, an additional crew member is required. Furthermore, an additional crew member is needed if two shifts that start at the same day are split.

6.2.3 Problem instance three

Train activities, which take place at two relief locations, are planned on two loclines. The same sequence of train activities is planned at each day of the week. A schematic representation of the train activities planned per day on each locline is shown in Figure 11. The locomotive allocated to L_1 is mobilized between 12 PM and 7:30 PM. The locomotive allocated to L_2 is mobilized between 00:20 PM and 7:50 PM.

The global drivers that are part of the employment group have the route knowledge, traction knowledge and skills for all 112 crew requirements. The maximum number of crew rosters that can be covered to the employment group is ten. A break room is present at the relief locations. Crew deadheading by foot takes 10 minutes between each pair of locations. The duration of a shift must be between 5 and 12 hours. The total duration of all train services and rolling stock deadheading operations included in a shift may not exceed 10 hours. The required meal break duration is 30 minutes, and this break can not be split into smaller breaks. The break is planned at a location with a break room present, and the break can start anywhere within the shift.

In the **optimal solution**, fourteen shifts are selected that must be covered by three employees. Each shift covers the crew requirements planned on the same locline at the same day. The required meal break is planned directly after the first shut-down activity in the shift. The objective value of the optimal solution is 6457.66 euro. The components of the solution value are specified by Equation (6.4).

$$\begin{aligned} z =& \sum_{s \in S}\Big(30(d_s - d_s^{break}) + c_s^{HLP} + 10d_s^{claim} + 5d_s^{npro} + 50a_s^{night}\Big)X_s \\ &+ \sum_{c \in C}\Big(10,000\mu_c U_c + 30d_c O_c\Big) + \sum_{d \in D}\Big(1000Y_d + max(36Y_d - \sum_{s \in S} a_{s,d}^2 d_s, 0)\Big) \\ =& 14x30x(8\frac{1}{12} - \frac{1}{2}) + 14x2x\frac{1}{3} + 10x1\frac{5}{6} + 5x14x3\frac{1}{2} + 50x0 \\ &+ 10,000x0 + 30x0 + 1000x3 + 0 \\ \approx& 6457.67 \textit{ euro} \end{aligned} \tag{6.4}$$

Each shift selected in the optimal solution has a duration of 8 hours and 5 minutes. The paid time is equal to the shift duration minus the required break duration of half an hour. The crew deadheading duration per shift is 20 minutes. The total shift duration on Friday after 7 PM is 1 hour and 50 minutes and the non-productive time per shift is 3 hours and 30 minutes. None of the selected shifts is a night shift. The average weekly workload per employee is roughly 37 hours and 40 minutes.

Figure 10: Schematic representation of the train activities planned on Monday till Saturday in problem instance two. Twenty train activities are planned on two locomotives. The locomotive allocated to L_1 is mobilized between 0:45 AM and 5:45 AM, and between 0:45 PM and 5:45 PM. The locomotive allocated to L_2 is mobilized between 6:15 AM and 11:15 AM, and between 6:15 PM and 11:15 PM

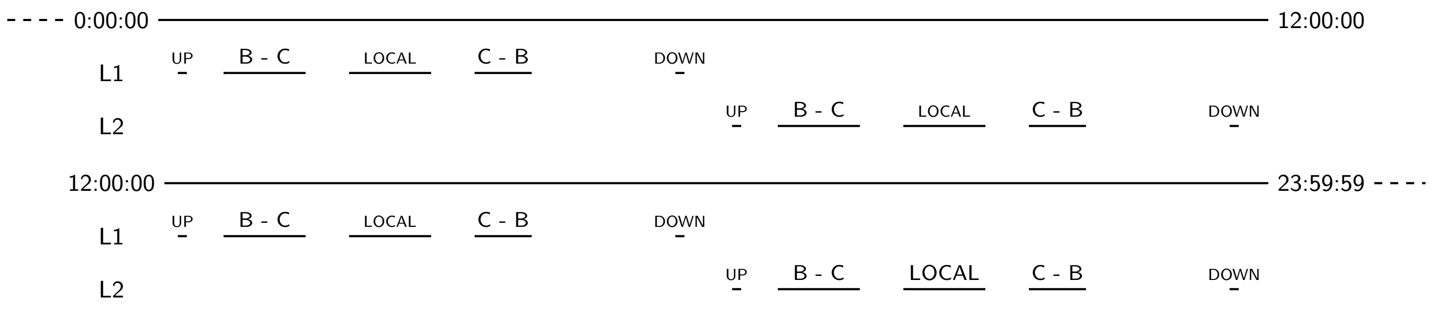
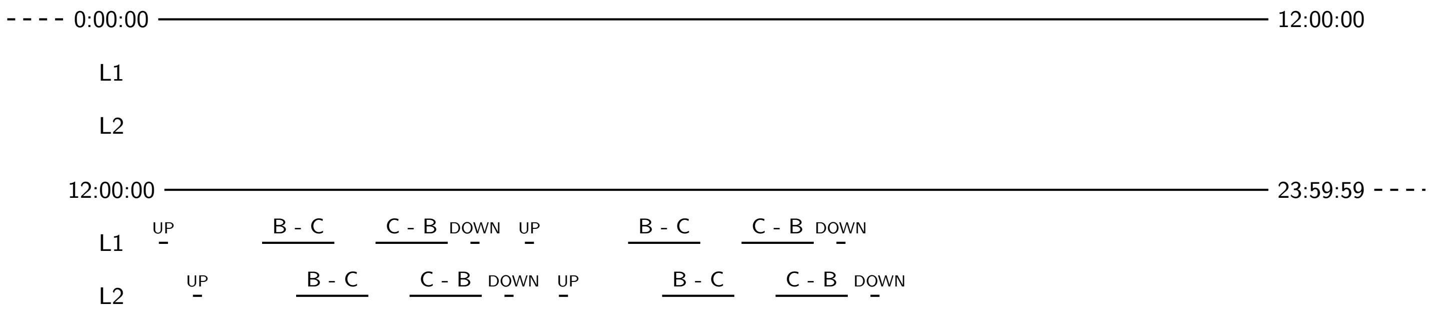


Figure 11: Schematic representation of the train activities planned per day in problem instances three and four. Sixteen train activities are planned on two loclines. The locomotive allocated to L_1 is mobilized between 12 PM and 7:30 PM. The locomotive allocated to L_2 is mobilized between 00:20 PM and 7:50 PM



6.2.4 Problem instance four

The only difference between problem instances three and four lies in the break requirements that are applicable to the crew. The minimum meal break duration is 20 minutes and the break may be split into two meal breaks with a duration of at least 10 minutes. Each break can be planned anywhere within the shift and crew repositioning may count as a break.

The optimal solution of problem instance three is also an **optimal solution** for this instance. However, the optimal solution is not unique. Crew repositioning from and to the home base of the employment group results in a sufficient break of 20 minutes in total. Therefore, each shift in the optimal solution consists of crew requirements that may be planned on different loclines. The spread times of the shifts selected in the optimal solution are not necessarily identical. The objective value of the optimal solution is 70 euros higher than the objective value of the optimal solution of problem instance three. This is caused by the fact that the required break duration is not paid, and the difference is $14 \times 30 \times (\frac{1}{2} - \frac{1}{3}) = 70$ euro.

6.3 Problem instance of the medium-size European rail-cargo company

The drivers of the European rail-cargo company are divided over 48 employment groups. Each driver must have knowledge of a certain part of the network to fulfill crew requirements. However, route knowledge is specified for only 40 employment groups. The other employment groups are removed from the data. The remaining drivers have the route knowledge to drive over each line segment that is used in the standard week. Only 34 employment groups with specified route knowledge have traction knowledge of at least one locomotive type used in the model week, and the other six employment groups are removed from the data. All crew members have the skills for all types of crew requirements. Employment groups with the same work function, work regime, skills, traction knowledge, route knowledge and home base location are merged. In the end, there are eleven employment groups with global driver and twelve employment groups with local drivers. Ten employment groups with global drivers have work regime R_1 and the other employment groups have work regime R_2. The crew depots are located in the default country. The average number of crew rosters that can be allocated to an employment group with global (local) drivers is 12.8 (17.8). The corresponding standard deviation is 6.058 (23.636).

The vehicle schedule consists of 143 loclines, which are included in 93 locline cycles. The number of train activities planned in the model week is 15,564. All crew requirements in a locline cycle can be performed by the same subset of employment groups, due to the fact that the same locomotive type is used to operate all loclines included in the cycle. All drivers who have work regime R_2 have the same route knowledge, traction knowledge and skills. As a result, every driver who have work regime R_2 can perform the same 7708 crew requirements, with a total duration of 219 hours and 30 minutes. The average duration of crew requirements that can be covered by drivers with work regime R_1 is 258 hours and 30 minutes, with a standard deviation of roughly 156 hours. The drivers of four employment groups with work regime R_1 have the traction knowledge to perform (almost) all crew requirements. Information about the driving operations (LEG), shut-down activities (DOWN) and local train activities (LOCAL) is given in Table 3. Recall that a shut-down secures the unit whilst unmanned, One crew requirement has a duration of more than 39 hours. This activity is marked as an outlier and removed from the data.

Table 3: Information about the train activities included in the real-life problem instance

Type	Number of crew requirements	Minimum duration	Maximum duration	Total duration
LEG	6660	0:01:00	7:17:00	266 days, 02:00:00
DOWN	1347	0:03:00	0:18:00	7 days, 01:02:00
LOCAL	7557	0:01:00	7:55:00	185 days, 11:14:00
Total	15,564	0:01:00	7:55:00	458 days, 14:16:00

After a global look at the data, two types of data inconsistencies were found. First of all, it occurs 45 times that the end location of a train activity is not equal to the start location of the next activity planned on the same locomotive. Secondly, some train activities planned in sequence on the same locomotive overlap in time. This happens two times, and the overlap time is 5 minutes. We assume that the corresponding crew requirements must be operated by the same employee, since no hand-over of responsibility can be scheduled between the crew requirements.

At most 112 train activities are scheduled at the same time. Due to the operational constraints that must be fulfilled by the crew roster, it is likely that $\sum_{d \in D} Y_d$ is larger than 112.

Table 4 shows the total duration of train activities planned at each day of the week. The workload on Saturday till Monday is relatively low in comparison with the workload on the other days. We expect that most shifts in the crew schedule start on Tuesday till Friday. Days off are mainly scheduled during the weekend and on Monday.

Table 4: The total duration of train activities planned at each day of the week

Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
60 days, 16:48:00	84 days, 14:10:00	83 days, 3:53:00	83 days, 12:46:00	82 days, 17:04:00	50 days, 9:53:00	13 days, 11:42:00

Meal breaks are planned according to the break requirements, which depend on the work function and work regime of the employment group to which the shift is allocated and on the shift characteristics. Detailed information about the break requirements can be found in Table A27 in Appendix

A. The minimum and maximum shift duration and the maximum driving time per shift also depend on the work function, work regime and shift type. More information about these restrictions can be found in Table A25 and Table A26 in Appendix A.

Table 5 provides information about the crew repositioning options that can be used by drivers of the European rail-cargo company. The longest direct trip with a regular taxi (not the expensive taxicab) between two locations in the same country takes 3 hours, and the maximum driving time between locations in different countries is 7 hours. If it is not possible to travel between a pair of locations within two deadheading operations, an expensive taxicab can be used. A trip with the expensive taxicab takes 3 hours if the locations are located in the same country, and 7 hours otherwise. A ride with the expensive taxicab costs 1000 euro, irrespective of the length of the trip.

Table 5: Information about the crew deadheading options that can be used by drivers of the European rail-cargo company

Option	**Number of segments**	**Minimum duration**	**Maximum duration**	**Cost (euro per hour)**
Foot	254	0:05:00	0:45:00	2
Lease car	207	0:10:00	2:25:00	20
Own car	147	0:05:00	3:20:00	15
Regular taxi	1272	0:01:00	7:00:00	25

There are six types of locations. Some neighboring locations are bundled and trucks are used to transport the cargo from the central bundle location to other locations in the bundle. The home base location of an employment group with global drivers is called a depot. Local drivers are connected to work seats. The other three location types are stations, work locations and work stations.

The number of locations in the original dataset is 1026. These locations are divided over 21 countries in Europe. There is a canteen present at roughly 62 percent of the bundle locations and 65 percent of the bundle locations is a relief location. There is no canteen present at non-relief locations. The locations that are relevant during crew scheduling are specified by the list below. More information about the relevant locations in the real-life problem instance is given in Table 6.

- Home base locations,
- Relief locations at which at least one train activity starts and/or ends,
- Non-relief locations at which at least one driving activity starts and/or ends that can be used to deadhead crew by train.

Table 6: Information about relevant locations included in the real-life problem instance, including the home base locations, the relief locations at which at least one train activity starts and/or ends, and locations at which at least one driving operation starts and/or ends

Location type	**Number of locations**	**Number of locations in default country**	**No break room and no relief location**	**No break room and relief location**	**Break room and relief location**
Bundle	333	241	47	14	272
Depot[1]	11	11	11	-	-
Station	6	6	6	-	-
Work location	1	1	1	-	-
Work station	14	14	14	-	-
Work seat[2]	12	12	12	-	-

[1] Home base locations of global drivers. [2] Home base location of local drivers.

6.4 Problem instances derived from the real-life problem instance

Datasets derived from the problem instance provided by the European rail-cargo company are used to evaluate the robustness of the method, i.e., these datasets are used for sensitivity analysis on the number of crewblocks constructed, the number of employment groups included in the data and the complexity of the break requirements that are applicable to the drivers.

Table 7 gives an overview of subsets of crew requirements obtained from the set of crew requirements in the original instance. Either all crew requirements planned in a specified locline cycle are included in the subset, or none of them is included. The crew requirements included in subset i are also included in subset $i+1$, for $i \in \{5, 6, 7\}$.

Table 7: Information about the crew requirements that are included in the subsets derived from the real-life problem instance

Subset	5	6	7	8
Number of loclines	3	6	13	23
Number of crew requirements	350	721	1443	2464
Total duration of crew requirements	10 days, 23:58:00	19 days, 14:46:00	40 days, 20:37:00	72 days, 3:03:00

Five problem instances are constructed for each subset of crew requirements specified in Table 7. The differences between the five cases are described in Table 8. Instance A is the basic instance, and then we can either add a break (in two different ways) and consider one or more employment groups.

Table 8: Differences between five problem instances constructed for each subset of crew requirements specified in Table 7

Instance	Number of employment groups	Break requirement
A	1	-
B	1	B_{easy}
C	1	Break requirements of European rail-cargo company
D	10	B_{easy}
E	10	Break requirements of European rail-cargo company

Only one employment group is considered in instance A, B and C. The drivers that are part of the employment group are global drivers, and have work regime R_1. At most 100 crew rosters can be allocated to the employment group. Crew rosters can be allocated to ten employment groups with global drivers and work regime R_1 in problem instance D and E. The number of crew rosters that can be allocated to each employment group are derived from the original dataset, and the total number of crew rosters that can be covered by the ten employment groups is 133. Every driver has the traction knowledge, route knowledge and skills to perform all crew requirements scheduled in the model week.

Break requirement B_{easy} is defined as follows. The required meal break duration is 30 minutes. It is allowed to split the break into two smaller meal breaks, each having a duration of at least 15 minutes. The break can start anywhere in the shift. Crew repositioning may count as a break and it is allowed to have a break on a mobilized locomotive. These break requirements apply to national and international shifts. The break requirements specified by the European rail-cargo company must hold in problem instance C and E, see Table A27 in Appendix A.

7 Computational experiments

In this chapter, results for the different problem instances as described in Chapter 6 are provided. The object oriented optimization software Quintiq is used to implement the methods described in Chapter 5. The MIP formulations and LP formulations are solved by using CPLEX 12.5.1, which is accessible via the Quintiq software package. A laptop with 4.0 GB RAM memory and an Intel(R) Core(TM) i7-2620M CPU @ 2.70GHz processor is used to perform the computational experiments. It turned out that the available memory on this laptop was insufficient for computational experiment with problem instances 7 and 8, and the real world application. These experiments are performed on the online server of Ab Ovo, which provides sufficient memory.

The remainder of this chapter is organized as follows. In Section 7.1, results are presented and evaluated for the small problem instances described in Section 6.2. Hereafter, Section 7.2 describes the results of the sensitivity analysis. We investigate the effect of different parameters, problem sizes and implementation options on the performance of the method.

7.1 Results for the problem instances with known optimal solution

In this section, the results for the problem instances as described in Section 6.2 will be discussed. The results will be compared between the different dominance rules.

The column generation algorithm is terminated if no variable with negative reduced cost is identified in the pricing problem. The MIP of the RMP is solved after each iteration of the solution process, such that we can regard the progress and tailing-off behavior of the method. The problem instances are small, and we decided to construct the set of crewblocks so that each element in this set consists of exactly one workblock. Furthermore, parameter ρ_2 is 100 in the hand-over planning method.

The RCSP algorithm can not guarantee that the optimal path from the source to the target is found when the heuristic dominance rule is turned on. It might occur that no variable with negative reduced cost is identified in the pricing problem, even though a resource feasible variable with negative reduced cost exists. Hence, the algorithm can terminate before the optimal solution of the master problem is found when the heuristic dominance rule is turned on. Recall that the master problem is the relaxed version of the original program. A resource feasible path from the source to the target with minimal reduced cost is identified when the strict dominance rule is turned on. When the strict dominance rule is turned on, the optimal solution of the master problem is found if all solution variables generated inside the pricing problem have non-negative reduced cost.

In theory, the feasible region of an LP relaxation is larger than the feasible region of the original IP formulation. This implies that the optimal solution of the master problem provides a lower bound on the optimal objective value of the original program. The algorithm can only guarantee that the optimal solution to crew scheduling is found if the optimal solution to the master problem is integral.

The integrality gap denotes the gap between the optimal linear solution and the optimal integer solution. Suppose that the objective function of the optimal solution of the original program is given by z^*. Let z^*_{LP} denotes the optimal solution of the master problem. The integrality gap is calculated using Equation (7.1).

$$\epsilon_{int} = \frac{z^* - z^*_{LP}}{z^*_{LP}} \times 100\% \tag{7.1}$$

The optimality gap denotes the gap between the optimal solution and the obtained solution. The integer solution obtained is given by z^H. The optimality gap is calculated using Equation (7.2).

$$\epsilon_{opt} = \frac{z^H - z^*}{z^*} \times 100\% \tag{7.2}$$

The optimal solution of the original program is not known for larger problem instances. Recall that the optimal solution of the master problem provides a lower bound on the optimal integer solution. An upper bound on the optimality gap between z^* and z^H is given when z^* is substituted by z^*_{LP} in Equation (7.1).

The candidate list of shifts constructed before the algorithm terminates might be different for independent runs. To evaluate the variance in the performance of the solution method, I different runs are performed. The best (worst) solution to the original program is given by $\underline{z}^I$ ($\bar{z}^I$). The average results over I independent runs are denoted by $\langle z \rangle^I$ (MIP objective value), $\langle Y \rangle^I$ (number of crew rosters) and $\langle d \rangle^I$ (total workduration covered). The average computation time in seconds after which the algorithm terminates is given by $\langle t \rangle^I$.

7.1.1 Results for problem instance one

The workblocks are constructed in the first stage of the process. There is a relief opportunity after all 84 crew requirement, and every workblock consists of exactly one crew requirement. The crewblocks are constructed in the second stage of the process. Every crewblock consists of exactly one workblock.

The algorithm is implemented such that the pricing subproblems are solved for the entire week. Two individual pricing subproblems are solved in each iteration of the process, one for each employment group. Both problems are stated on an acyclic directed graph with 214 nodes and 677 arcs. We run the algorithm in series. The average results over ten independent runs are shown in Table 9.

The optimal solution to the master problem is found in each run, and z^*_{LP} is 6898.50 euro. The integrality gap is 2.9%. The optimal solution value of the original program is 7100.33 euro. From Table 9, we can calculate that the tightest upper bound on the optimality gap is 4.6% (strict). The

Table 9: Results for problem instance one over ten independent runs. The standard deviations are listed between brackets.

Dominance rule	$\underline{z}^{10}$	$\langle z \rangle^{10}$	$\overline{z}^{10}$	$\langle t \rangle^{10}$	$\langle Y \rangle^{10}$	$\langle d \rangle^{10}$
Heuristic	7563.25	8132.28 (209.341)	8278.67	17.1 (2.745)	3.3 (0.458)	2 days, 23:07:21 (0:02:27)
Strict	7425.50	7835.96 (389.328)	8512.82	17.7 (6.573)	3.3 (0.458)	2 days, 23:09:00 (0:02:00)

optimal integer solution is not found in any run. We can conclude that the strict dominance rule outperforms the heuristic dominance rule on average for this problem instance.

As can be seen from Table 9, not all crew requirements are always covered by at least one selected shift. At most one crew requirement with a duration of 5 minutes is not assigned, and the corresponding penalty cost is 10,000 / 12 $\approx$ 833.33 euro. At least one shift in the candidate list contains the unassigned crew requirement, given that the optimal LP solution is found. The penalty cost is smaller than the additional cost associated with the selection of one of the shifts covering the unassigned crew requirement. An educated guess is that this problem is solved if θ is increased drastically. For example, when θ is set to 100,000.

The unequal variance Student's t-test (Student's t-test or t-test in the following) is used to evaluate whether the difference in runtime for the two dominance rules is significant. We refer the interested reader to Appendix B for more information about the Student's t-test . The first (second) sample consists of the computation times of the independent runs when the heuristic (strict) dominance rule is turned on. The corresponding p-value is 0.711. Hence, the null hypothesis of equal sample means can not be rejected at a significance level of 5%.

Recall that the CSP is easier and faster to solve if multiple workblocks are combined into one crewblock in the second stage of the process. The optimal solution of problem instance one can be attained if the elementary sequence of crew requirements planned on one locline at the same day is included in one or more crewblocks. If crewblocks are constructed with the method described in Subsection 5.1.1, the workblocks are combined into 28 crewblocks. It takes roughly 3.9 seconds to solve the LP formulation to optimality, irrespective of the dominance rule that is turned on. The optimal solution to crew scheduling was found in all twenty runs. If all crew requirements planned on one locline at the same day are directly included in one crewblock, the solution time is approximately 2.6 seconds. Each shift selected in the optimal solution covers exactly one crewblock, and the corresponding objective value is 7100.33.

The explicit consideration of all paths takes more than 5 minutes, given that each crewblock consists of one crew requirement. If we run the algorithm with the dominance rule off, all paths are considered and that is potentially very slow. We can speed up the process by limiting the number of feasible paths after which the algorithm terminates. Unfortunately, the algorithm can not guarantee that the optimal solution is in one of the found paths up to the point where it stops. Let us consider the case in which only the first ten feasible paths are returned by each pricing subproblem. The average objective value over ten independent runs is approximately 162,300 after 1 minute of runtime. Performance improvements are obtained by setting a dominance rule. We do not consider the option to turn the dominance rule off in the remainder of this thesis.

7.1.2 Results for problem instance two

The workblocks are constructed in the first stage of the process. There is a relief opportunity after all 120 crew requirements planned in the model week, and every workblock consists of exactly one crew requirement. The crewblocks are constructed in the second stage of the process. Every crewblock consists of exactly one workblock.

The algorithm is implemented such that one pricing problem is solved for the entire week. The pricing problem is stated on an acyclic directed graph with 526 nodes and 1477 arcs. Table 10 summarizes the average results over ten independent runs.

Table 10: Results for problem instance two over ten independent runs. The standard deviations are listed between brackets.

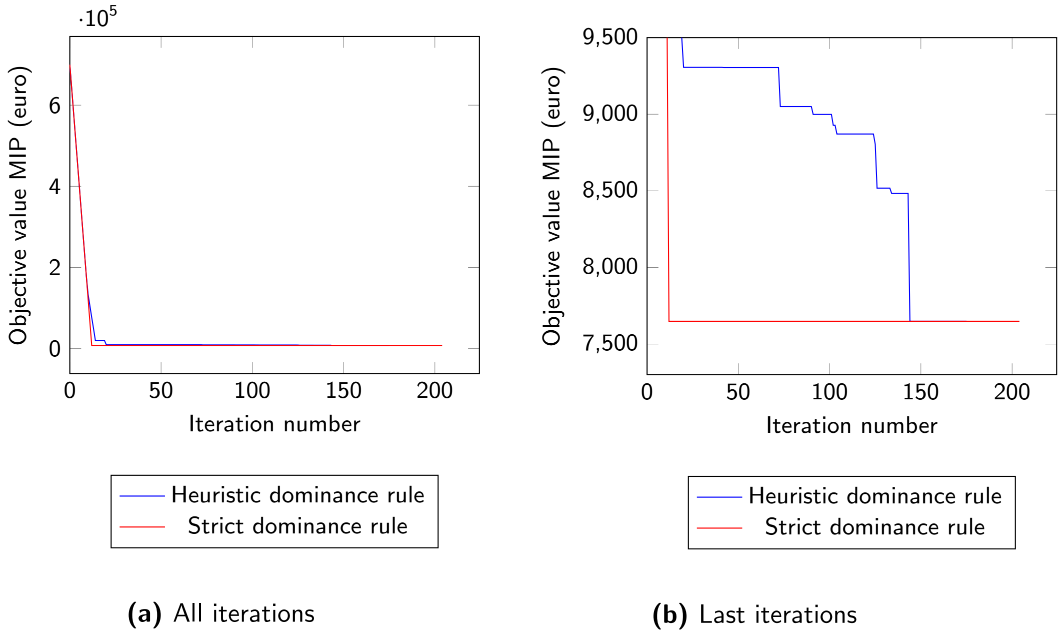
Dominance rule	$\underline{z}^{10}$	$\langle z\rangle^{10}$	$\bar{z}^{10}$	$\langle t\rangle^{10}$	$\langle Y\rangle^{10}$	$\langle d\rangle^{10}$
Heuristic	7649.17	7653.88 (9.944)	7672,75	36.9 10.289)	3	2 days, 22:00:00
Strict	7649.17	7649.17	7649.17	75.4 (14.659)	3	2 days, 22:00:00

From Table 10, we can conclude that the optimal solution to crew scheduling is found in all runs if the strict dominance rule is turned on. The objective value of the optimal LP solution is 7621.02 and the average number of shifts constructed per run is about 206. The integrality gap is 0.4%.

The optimal solution of the original program is not found in two out of ten runs, given that the heuristic dominance rule is turned on. Furthermore, the optimal solution of the master problem was not found in six independent runs (objective values between 7622.22 and 7639.10). The algorithm terminates before the optimal LP solution is found. In the last iteration of the process, the algorithm was not able to identify a variable with negative reduced cost in the pricing problem. One way to solve this problem is to consider a combination of dominance rules. First, the heuristic dominance rule is turned on. Promising shifts are constructed as long as a path with negative reduced cost is identified in the pricing problem. Afterwards, the strict dominance rule is turned on, and promising shifts are identified with the current dual prices.

We will now compare the results between different dominance rules. First, one representation run in which the optimal LP solution is found is selected for both dominance rules. Figure 12 shows a plot of the objective value of the MIP of the RMP versus the iteration number. Figure 12a is a plot of the objective value versus the iteration number from the start of the solution process until no variable with negative reduced cost is identified in the pricing problem. Figure 12b zooms in on later iterations.

Figure 12: Objective value of the original program, i.e., the MIP of the RMP, expended to the iteration number

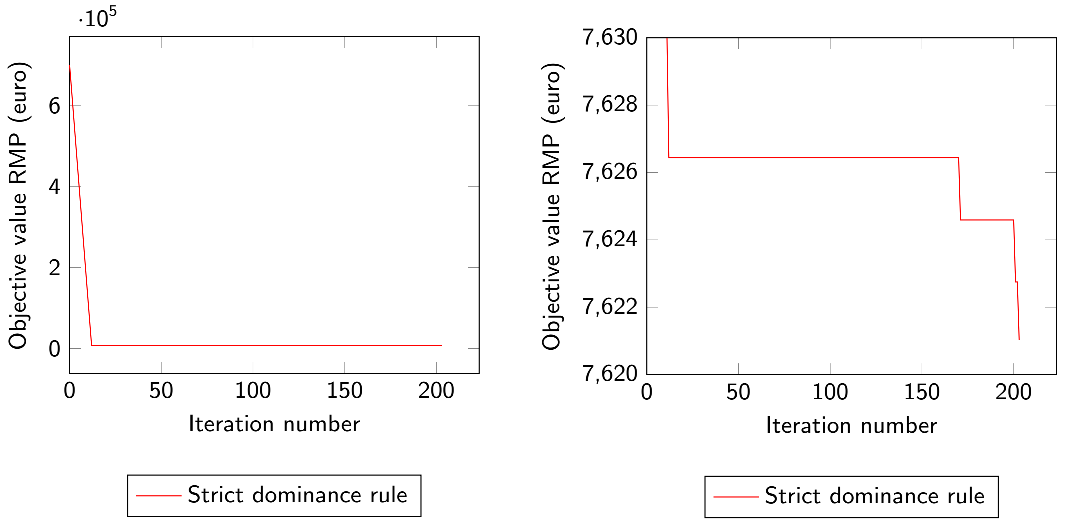


(a) All iterations **(b)** Last iterations

At the start of the solution process, the candidate list of shifts is empty and the corresponding objective value of (4.1) - (4.15) is 700,000 euro. The convergence of the column generation method is fast at the start of the solution process, since promising shifts are constructed that contain crewblocks which are not yet covered by the crew schedule (see Figure 12a). Figure 12b shows that the optimal solution of the original program is found after twelve iterations if the strict dominance rule is turned on, and after 144 iterations otherwise. Furthermore, the number of iterations needed before the column generation process terminates exceeds 200 if the strict dominance rule is turned on, while it is only 175 if the heuristic dominance rule is turned on. Recall that the RCSP can not guarantee that a resource feasible path with minimal reduced cost is found in the pricing problem if the heuristic dominance rule is turned on.

The tailing-off behavior of the column generation technique is visible by the red line in Figure 13. At the beginning of the process, the convergence of the column generation method is fast. The objective value of the RMP decreases rather slow near the end of the solution process, see Figure 13b. The objective value of the RMP decreases roughly 692.374 euro in the first twelve iterations, and only 6 euro in the remaining 191 iterations. Moreover, the objective value of the MIP of the RMP did not improve in those 191 iterations (see Figure 12b). These results indicate that the overall performance of the solution method might be improved by terminating the column generation algorithm if the objective value of the (MIP of the) RMP did not improve in a predefined number of iterations.

Figure 13: Objective value of the RMP expended to the iteration number



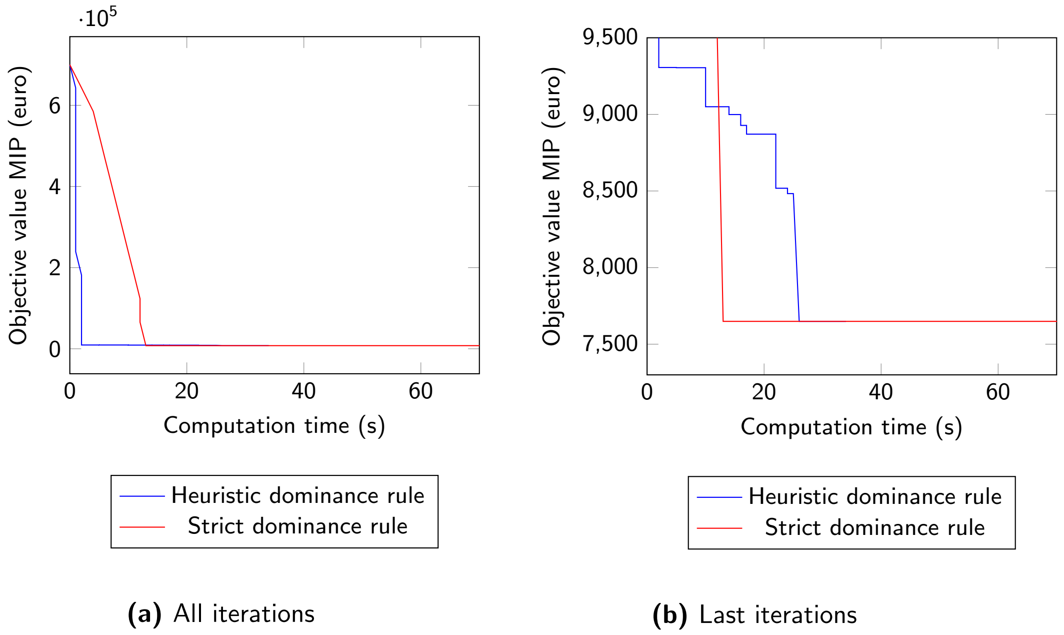
(a) All iterations

(b) Last iterations

Figure 14 shows a plot of the objective value of the MIP of the RMP versus the computation time, for the two representative runs. Figure 14a is a plot of the objective value versus the computation time from the start of the solution process until no variable with negative reduced cost is identified in the pricing problem. Figure 14b zooms in on later iterations.

Since the slope of the blue line (heuristic) in Figure 14a is steeper than the slope of the red line (strict), we can conclude that the objective value of the MIP formulation decreases fastest if the heuristic dominance rule is turned on. On the other hand, we can conclude from Figure 14b that the runtime to find the optimal MIP solution is shorter if the strict dominance rule is turned

Figure 14: Objective value of the original program, i.e., the MIP of the RMP, expended to the computation time



(a) All iterations **(b)** Last iterations

on. These results reflect the characteristics of the dominance rules. All or almost all paths must be considered to find the resource feasible path with minimal reduced cost when the strict dominance rule is turned on. Significantly fewer paths are considered when the heuristic dominance rule is turned on. Promising shifts are identified in the pricing problem, that are not necessary optimal.

There are a number of implementation options that can improve the performance of the solution approach. First of all, there is no need to solve the MIP of the RMP until the last iteration of the solution process, given that the algorithm only terminates if no variable with negative reduced cost is identified in the pricing problem. Secondly, the algorithm can be implemented such that the pricing subproblems are split per day of the week. The resulting pricing subproblems are easier and faster to solve, as the number of resource feasible paths in the graph decreases. Finally, the algorithm can be implemented such that the individual pricing subproblems are solved in parallel. The number of threads depends on the number of CPU cores available.

The Student's t-test is used to evaluate whether the average running time is reduced significantly if one or more of these implementation options are used. Table 11 shows the average computation time and sample variance for each (combination of) implementation option(s) over fifty independent runs in which the strict dominance rule is turned on. All the reported running times are obtained by solving the master problem to optimality. As a result, the running times are comparable with each other.

Table 11: The sample mean (average computation time) and sample variance over fifty independent runs in which the strict dominance rule is turned on.

Sample	MIP only at end	Pricing problem per day	Algorithm in parallel	$\langle t \rangle^{50}$	s^2
1	FALSE	FALSE	FALSE	68.72	130.451
2	TRUE	FALSE	FALSE	49.88	43.536
3	FALSE	TRUE	FALSE	19.40	8.816
4	FALSE	FALSE	TRUE	67.93	128.672
5	TRUE	TRUE	FALSE	15.00	4.449
6	FALSE	TRUE	TRUE	14.94	5.323
7	TRUE	FALSE	TRUE	50.02	44.218
8	TRUE	TRUE	TRUE	10.82	2.477

If we perform the statistical hypothesis test on Sample 1 and Sample 4, the null hypothesis of equal means can not be rejected. This is caused by the fact that only one pricing problem is solved in each iteration of the process. The same conclusion is drawn if we perform the test on Sample 2 and Sample 7. The average computation time is reduced significantly if the MIP of the RMP is only solved in the last iteration or if the pricing problems are solved per day of the week. Combining two of the implementation options leads to a significant reduction of the runtime. We can not reject the null hypothesis of equal sample means if we perform the t-test on Sample 5 and Sample 6. The average computation time is reduced significantly if all three implementation options are combined.

We can conclude that it is computationally attractive to split the pricing problems per day of the week. The resulting pricing subproblems can be solved in parallel to speed up the process. Furthermore, the running time is reduced significantly if the MIP of the RMP is not solved in each iteration of the solution process.

7.1.3 Results for problem instances three and four

The workblocks are constructed in the first stage of the process. There is a relief opportunity after all 112 crew requirement planned in the model week, and every workblock consists of exactly one crew requirement. The crewblocks are constructed in the second stage of the process. Every crewblock consists of exactly one workblock.

One pricing problem is solved in each iteration of the process if the CSP of problem instance three is solved. The pricing problem is stated on an acyclic directed graph with 1238 nodes and 4114 arcs. This graph is large in comparison with the graphs on which the pricing subproblems of problem instances one and two are stated, due to the break requirements. The average results over ten independent runs for problem instance three are defined in Table 12.

Table 12: Results for problem instance three over ten independent runs. The standard deviations are listed between brackets, if larger than zero.

Dominance rule	$\underline{z}^{10}$	$\langle z \rangle^{10}$	$\bar{z}^{10}$	$\langle t \rangle^{10}$	$\langle Y \rangle^{10}$	$\langle d \rangle^{10}$
Heuristic	8579.42	18,827.61 (12,640.36)	38,563.28	34.8 (8.784)	4	2 days, 11:38:30 (1:17:08)
Strict	6457.67	6457.67	6457.67	126.0 (19.303)	3	2 days, 12:40:00

The optimal solution to the master problem is found in each run when the strict dominance rule is turned on, and z^*_{LP} is 6257.67 euro. The integrality gap is 3.2%. The master problem is not solved to optimality in any of the ten independent runs in which the heuristic dominance rule is turned on. Recall that the optimal solution value of the original program is 6457.67 euro. As can be seen from Table 12, the optimal solution of the original problem is found in all ten runs if the strict dominance rule is turned on. The average number of shifts constructed before the optimal LP solution was found is 325 and the corresponding standard deviation is approximately 32. From Table 12, we can calculate that the smallest optimality gap is 37.1% when the heuristic dominance rule is turned on. We can conclude that the strict dominance rule outperforms the heuristic dominance rule for this problem instance.

The pricing problem of problem instance four is stated on an acyclic directed graph with 1238 nodes and 4149 arcs. If we compare the size of the graph with the graph constructed for pricing problem three, we can conclude that the number of arcs is increased. It is worth mentioning that the difference is only very small. There are multiple reasons for this. First, the problem instance is relatively small. Second, the minimum duration per break decreases from 30 minutes to only 10 minutes. The increase in the number of arcs with a break outweighs the decrease in the number of wait arcs only minimal. The average results over ten independent runs for the fourth problem instance are given in Table 13.

Table 13: Results for problem instance four over ten independent runs. The standard deviations are listed between brackets, if larger than zero.

Dominance rule	$\underline{z}^{10}$	$\langle z \rangle^{10}$	$\bar{z}^{10}$	$\langle t \rangle^{10}$	$\langle Y \rangle^{10}$	$\langle d \rangle^{10}$
Heuristic	6527.67	6532.58 (2.70)	6535.42	47.2 (8.623)	3	2 days, 12:40:00
Strict	6527.67	6532.58 (2.93)	6538,00	46.1 (8.264)	3	2 days, 12:40:00

The optimal LP solution of 6327.67 is attained in each run, irrespective of the dominance rule used. The integrality gap is 3.2%. The optimal solution to crew scheduling is found in only two runs, but the optimality gap is at most 0.1 percent. The Student's t-test is used to evaluate whether the difference between the average runtime for both dominance rules is significant. The p-value of the t-test is 0.785. We fail to reject the null hypothesis of equal sample means at a significance level of 5%.

7.2 Sensitivity analysis

Consider the original dataset provided by the European rail-cargo company. 483 directed acyclic graphs must be initialized when the pricing subproblems are split per day of the week. Only 135 of these graphs are initialized after 4 hours. By then, the total number of nodes is roughly 650,000. Approximately 28,500,000 arcs are constructed between these nodes. We conclude from these results that the solution approach as described in Chapter 5 can not solve the CSP of the medium-size European rail-cargo company within a reasonable time limit.

Recall that problem instances are derived from the original problem to evaluate the robustness of the method to different implementation options, parameters and data sizes. In Subsection 7.2.1 we evaluate the crewblock construction methods that are described in Section 5.1. Furthermore, we investigate the effect of the parameter settings in the hand-over plannings method. Recall that two different dominance rules can be turned on. In Subsection 7.2.2 we evaluate each (combination of) dominance rule(s). The performance improvements obtained by splitting the pricing subproblems on weekday are described in Subsection 7.2.3. The effect of the number of crewblocks on the solution is investigated in Subsection 7.2.4. In Subsection 7.2.5 we investigate how sensitive the model reacts to an increase in the number of crew requirements and the complexity of the break rules. Finally, in Subsection 7.2.6 the effect of the number of employees on the solution is investigated.

The convergence of the column generation method is rather slow near the end of the solution process, meaning that the objective value of the (MIP of the) RMP does not decrease much in the last iterations. To avoid tailing-off behavior, the column generation process is terminated if the objective value of the original program has not decreased by more than 10 euro in the last k iterations, where k is a predefined parameter. Furthermore, the column generation process is terminated if the covered work duration and number of employees needed have not changed in the last m iterations. We ran tests using different values for these parameters, to obtain different computation times and solutions of varying qualities. Parameter k was set to either 20 or ∞ and m was set to either 50 or ∞. We can only guarantee that the optimal solution of the LP relaxation of (4.1) - (4.15) is found if both k and m are set to infinity, the strict dominance rule is turned on, and no variable with negative reduced cost can be identified in the pricing problem.

It takes some non-negligible amount of time to solve the MIP formulation of the RMP to optimality, given the set of shifts added to the formulation. It is reasonable that the solution time increases with the number of columns. The frequency with which the MIP must be solved depends on the values of k and m. There is no need to solve the MIP formulation of the RMP until the last iteration if both k and m are set to infinity. Only the solution in the last iteration is relevant.

7.2.1 Crewblock construction

The aim of this subsection is to evaluate the different crewblock construction methods described in Section 5.1. Due to time limitations, we only consider the crewblock construction methods described in Subsection 5.1.1 and Subsection 5.1.3.

Recall that crewblock $c \in C$ is represented by the sequence of crew requirements included in the crewblock, i.e. $c = (r_{1,c}, ..., r_{n_c,c})$ where $r_{i,c} \in R$ for all positions $i \in \{1, ..., n_c\}$ and n_c is the number of crew requirements included in crewblock c. The time elapsed between the start of $r_{1,c}$ and the end of $r_{n_c,c}$ is given by δ_c, for all $c \in C$. The duration of crewblock $c \in C$ is denoted by d_c. Table 14 summarizes the results of the crewblock construction method described in Subsection 5.1.1 (hereafter CCM-1), given that parameter ρ_2 is 100 in the hand-over planning method.

Table 14: Results of the crewblock construction method described in Subsection 5.1.1 (CCM-1). Parameter ρ_2 is equal to 100. The standard deviations are listed between brackets.

Dataset	$\lvert C\rvert$	$\sum_{c\in C} \delta_c$	$\sum_{c\in C} \delta_c$ / $\lvert C\rvert$	$\sum_{c\in C} d_c$	$\sum_{c\in C} d_c$ / $\lvert C\rvert$
5	71	11 days, 10:14:00	3:51:45 (1:09:53)	12 days, 8:31:00	4:10:35 (1:13:50)
6	133	21 days, 14:55:00	3:54:06 (1:25:31)	22 days, 23:41:00	4:08:53 (1:28:19)
7	298	46 days, 00:19:00	3:42:21 (2:00:25)	49 days, 1:34:00	3:57:06 (2:02:01)
8	512	80 days, 8:04:00	3:45:57 (1:48:37)	89 days, 14:54:00	4:12:04 (2:21:53)

We will now describe the results obtained for the crewblock construction method described in Subsection 5.1.3. In that method, δ_c is enforce to be in the interval $[\underline{T}, \bar{T})$, for all $c \in C$. Let C_1 be the set of crewblocks $c \in C$ for which $\delta_c < \underline{T}$ and C_2 the set of crewblock $c \in C$ for which $\delta_c > \bar{T}$. Table 15 summarizes the effect of an increase or decrease of $\underline{T}$ and $\bar{T}$ on $|C|, |C_1|$ and $|C_2|$. We must note that it might occur that one or more of these numbers do not change as a result of an increase or decrease of $\underline{T}$ and $\bar{T}$.

Table 15: Effect of an increase or decrease of the minimum and maximum crewblock length on the number of crewblocks and the number of crewblocks that violate the duration constraints

	$\lvert C\rvert$	$\lvert C_1\rvert$	$\lvert C_2\rvert$
$\underline{T}\uparrow$, $\bar{T}\uparrow$	$\downarrow$	$\downarrow$ or $\uparrow$	$\downarrow$
$\underline{T}\uparrow$, $\bar{T}\downarrow$	$\downarrow$ or $\uparrow$	$\uparrow$	$\uparrow$
$\underline{T}\downarrow$, $\bar{T}\uparrow$	$\downarrow$ or $\uparrow$	$\downarrow$	$\downarrow$
$\underline{T}\downarrow$, $\bar{T}\downarrow$	$\uparrow$	$\downarrow$ or $\uparrow$	$\uparrow$

Table 16 summarizes the results of the crewblock construction method obtained for problem instance 5, for different combinations of $\underline{T}$ and $\bar{T}$. The number of improvement iterations is set to zero, as the same crewblocks must be constructed in each independent run to make a fair comparison between different implementation options, parameter settings and data sizes. Furthermore, it turned out that only small improvements were realized if the local search heuristic is applied. The maximum idle time between two consecutively scheduled crew requirements included in the same crewblock ($\bar{M}$) is 3 hours. Two consecutively planned crew requirements are included in different workblocks if the idle time between these activities is greater than or equal to the hand-over duration of 1 minute and if the end location of the first activity is a relief location, i.e., $\underline{M}$ is 1 minute. The hand-over activity required after crewblock c is planned directly before the start of crew requirement r_{1,c_c^+}, for all $c \in C$ that do not end with a shut-down activity.

Table 16: Results of the crewblock construction method described in Subsection 5.1.3 for problem instance 5. Parameter ρ_2 is equal to 100. The standard deviations are listed between brackets.

$\underline{T}$	$\bar{T}$	$\lvert C\rvert$	$\lvert C_1\rvert$	$\lvert C_2\rvert$	$\sum_{c\in C}\delta_c$	$\sum_{c\in C}\delta_c$ / $\lvert C\rvert$	$\sum_{c\in C} d_c$	$\sum_{c\in C} d_c$ / $\lvert C\rvert$
2	5	80	10	2	11 days, 9:44:00	3:25:18 (1:18:45)	12 days, 9:01:00	3:42:46 (1:30:29)
3	5	80	19	2	11 days, 12:58:00	3:27:44 (1:18:13)	12 days, 11:36:00	3:44:42 (1:30:08)
1	6	83	0	0	11 days, 7:51:00	3:16:31 (1:10:48)	12 days, 9:04:00	3:34:45 (1:23:03)
2	6	72	2	0	11 days, 9:08:00	3:47:37 (1:07:30)	12 days, 8:53:00	4:07:24 (1:17:19)
0	7	141	0	0	11 days, 1:17:00	1:52:53 (1:36:56)	12 days, 9:41:00	2:06:40 (1:47:53)
1	7	83	0	0	11 days, 7:51:00	3:16:31 (1:10:48)	12 days, 9:04:00	3:34:45 (1:23:03)
2	7	70	0	0	11 days, 10:02:00	3:54:53 (1:10:05)	12 days, 8:51:00	4:14:27 (1:34:12)

From Table 16, we can conclude that the number of crewblock in C does not increase with $\underline{T}$. This is in caused by the fact that more workblocks are combined into one crewblock when $\underline{T}$ increases, since the size of C_1 must be minimized. For a given value of $\bar{T}$, the number of elements in C_1 increases with $\underline{T}$. Two crewblocks that consist of only one workblock have a corresponding δ_c

between 5 hours (including) and 6 hours (excluding), since $|C_2| = 2$ if $\bar{T} = 5$ hours and $|C_2| = 0$ is $\bar{T} = 6$ hours.

Two combinations of $\underline{T}$ and $\bar{T}$ are selected to evaluate the effect of the number of crewblocks on the solution quality and corresponding computation time. The first combination is $\underline{T} = 1$ hour and $\bar{T} = 7$ hours, and the second combination is $\underline{T} = 0$ hour and $\bar{T} = 7$ hours (hereafter CCM-2 and CCM-3, respectively). The number of nodes in each pricing subproblem is reduced if multiple workblocks are merged into one crewblock, and also the number of arcs decreases (more drastically). As a result, the CSP is easier and faster to solve. However, there is a trade-off between the size of the CSP and the optimal solution value. In fact, the solution space is reduced if multiple workblocks are combined into one crewblock, and some shifts that can be generated if each crewblock consists of one workblock can not be constructed after merging. Each crewblock consists of exactly one workblock if $\underline{T} = 0$. Each workblock is a sequence of crew requirements that must be assigned to the same driver. Table 17 summarizes the results of CCM-2 and CCM-3, given that parameter ρ_2 is 100 in the hand-over planning method.

Table 17: Results of the crewblock construction methods CCM-2 and CCM-3. Parameter ρ_2 is equal to 100. The standard deviations are listed between brackets.

Method	**Dataset**	$\lvert C\rvert$	$\sum_{c \in C} \delta_c$	$\sum_{c \in C} \delta_c$ / $\lvert C\rvert$	$\sum_{c \in C} d_c$	$\sum_{c \in C} d_c$ / $\lvert C\rvert$
CCM-2	5	83	11 days, 7:51:00	3:16:31 (1:10:48)	12 days, 9:04:00	3:34:45 (1:23:03)
	6	185	20 days, 22:46:00	2:43:04 (1:21:11)	23 days, 5:04:00	3:00:40 (1:27:26)
	7	381	44 days, 13:22:00	2:48:24 (1:45:59)	49 days, 14:52:00	3:07:32 (1:48:09)
	8	649	77 days, 13:27:00	2:52:05 (1:33:49)	90 days, 9:02:00	3:20:32 (2:14:33)
CCM-3	5	141	11 days, 1:17:00	1:52:53 (1:36:56)	12 days, 9:41:00	2:06:40 (1:47:53)
	6	343	19 days, 18:29:00	1:23:00 (1:24:46)	23 days, 3:11:00	1:37:07 (1:30:28)
	7	743	41 days, 5:41:00	1:19:56 (1:38:31)	49 days, 8:59:00	1:34:52 (1:41:04)
	8	1205	72 days, 12:31:00	1:26:40 (1:34:47)	90 days, 2:27:00	1:47:40 (1:59:36)

Recall that the parameters ρ_1, ρ_2 and ρ_3 determine the start time of a crew change on a mobilized locomotive. The effect of the parameters ρ_1, ρ_2 and ρ_3 on the performance of the column generation approach is evaluated for problem instance 5C. The pricing problem is split by employment group and break requirement, and the resulting number of subproblems is three. Note that we do not split the pricing subproblems by day of the week. The size of the graph on which pricing subproblem $q \in Q$ is stated depends on the parameter values and the crewblock construction method used. Let V be the set with all nodes and A the set with all arcs, i.e., $V = \cup_{q \in Q} V^q$ and $A = \cup_{q \in Q} A^q$. The number of elements in V and A vary over different runs if $\rho_3 > 0$. Table 18 provides information about the number of elements in V and A, for different parameter settings and crewblock construction methods.

Table 18: The number of elements in V and A for problem instance 5C, split by crewblock construction method and parameter settings. The average result over ten different runs is given if $\rho_3 > 0$

Method	$\rho_1 = 100, \rho_2 = \rho_3 = 0$		$\rho_2 = 100, \rho_1 = \rho_3 = 0$		$\rho_3 = 100, \rho_1 = \rho_2 = 0$	
	$\lvert V\rvert$	$\lvert A\rvert$	$\lvert V\rvert$	$\lvert A\rvert$	$\lvert V\rvert$	$\lvert A\rvert$
CCM-1	4588	30,758	4606	30,862	4613.2 (9.859)	30,886.8 (43.814)
CCM-2	4830	31,975	4830	32,041	4830	31,983.4 (36.087)
CCM-3	5886	37,788	5898	37,940	5898	37,855.8 (44.381)

From Table 18, we conclude that the total number of nodes and arcs differ only marginal if other parameter values are set. Further, the number of nodes and arcs grows with the number of crewblocks in C. We will now investigate the effect of ρ_1, ρ_2 and ρ_3 on the performance of the column generation approach. The algorithm is implemented such that independent pricing subproblems are solved in parallel and the strict dominance rule is turned on. Further, k is 20 and m is 50. Table 19 shows the average results over ten independent runs for problem instance 5C. The optimal solution of the master problem is given by z^*_{LP}. The optimal LP solution is different from run to run if $\rho_3 > 0$.

Table 19: Results of solving the crew scheduling problem of dataset 5C for different combinations of parameter values and crewblock construction method

Method	ρ_1	ρ_2	ρ_3	z^*_{LP}	$\underline{z}^{10}$	$\langle z\rangle^{10}$	$\bar{z}^{10}$	$\langle t\rangle^{10}$
CCM-1	100	0	0	30,749.06	31,527.47	31,527.47	31,527.47	331.2 (19.39)
	0	100	0	30,376.99	31,258.05	31,258.05	31,258.05	321.3 (17.23)
	0	0	100	-	31,395.07	60,034.41 (25,227.70)	78,603.08	213.9 (14.91)
CCM-2	100	0	0	30,209.15	31,188.63	31,188.63	31,188.63	525.7 (34.67)
	0	100	0	30,002.68	30,191.30	30,191.30	30,191.30	537.1 (34.02)
	0	0	100	-	31,221.05	59,447.78 (25,711.28)	78,461.45	338.5 (20.52)
CCM-3	100	0	0	29,425.12	29,961.30	30,933.29 (846.450)	31,508.18	2874.0 (41.33)
	0	100	0	29,131,70	29,733.76	30,859.88 (827.516)	31,401.27	2898.8 (45.82)
	0	0	100	-	31,715.40	70,378.02 (21,614.66)	80,285.56	2042.3 (42.19)

The optimal LP solution is attained in all runs if $\rho_1 = 100$ or $\rho_2 = 100$, and crewblocks are constructed with CCM-1 or CCM-2. If crewblocks are constructed with CCM-3, each crewblock is covered at least once in all ten independent runs if $\rho_1 = 100$ or $\rho_2 = 100$. It is not always possible to cover all crewblocks if hand-over activities are planned randomly, irrespective of the construction method used. A crewblock can not be covered if its duration is too long. On the other hand, it might occur that it is not possible to plan a valid meal break in the shift that covers the crewblock.

Given the crewblock construction method, the optimal LP value is the lowest if hand-over activities are planned directly before the start of the crewblocks. However, the differences are only marginal and other conclusions must be draw for several other datasets. Our main goal is to evaluate the effect of different implementation options on the solution quality and computation time. Therefore, it is sufficient to chose one parameter setting. We have chosen to set ρ_2 equal to 100.

7.2.2 Effect of the dominance rule

Recall that the RCSP algorithm uses the concept of dominance rules to limit the number of paths that must be considered. In this thesis, we use the heuristic and the strict dominance rule as described in Paragraph 5.3.3.2. The aim of this subsection is to investigate the effect of both dominance rules on the performance of the method. We also investigate the effect of using combinations of dominance rules. First, the heuristic (strict) dominance rule is turned on. If the algorithm terminates before the optimal LP solution is found, the strict (heuristic) dominance rule is turned on. Additional paths are generated in the pricing problem until the algorithm terminates again. It is worth mentioning that the combination strict dominance followed by heuristic dominance can only improve the solution quality if at least one of the parameters *k* and *m* is not set to infinity, and the algorithm did not terminate due to the fact that no shift with negative reduced cost was identified in the pricing problem. Otherwise, the optimal solution of the master problem is already found. In that case, no shifts with negative reduced cost can be identified in the pricing problem of the RCSP algorithm by turning the heuristic dominance rule on. In this section, we consider problem instances 5A and 5C.

The result of computational experiments for instance 5A are given in Table 20. Further, *k* is 20 and *m* is 50. The MIP of the RMP is solved after each iteration.

Recall that turning the strict dominance rule on will find the path with the lowest reduced cost that is feasible with respect to the resources, but may have to consider all or almost all paths. Significantly fewer paths are traversed if the heuristic dominance rule is turned on, but the optimal path is not necessarily found. There is a trade-off between the solution quality and corresponding computation time, which can also be seen in Table 20.

The unequal variance Student's t-test is used to evaluate whether or not the average objective value is significantly different if another (combination of) dominance rule(s) is used. It turns out that the average objective value is significantly higher if the heuristic dominance rule is turned on than when the strict dominance rule or a combination of dominance rules is used, irrespective of

Table 20: The average result for problem instance 5A over ten independent runs for different (combinations of) dominance rules.

Method	**Heuristic dominance rule**		**Strict dominance rule**	
	$\langle z \rangle^{10}$	$\langle t \rangle^{10}$	$\langle z \rangle^{10}$	$\langle t \rangle^{10}$
CCM-1	196,205.64 (59,635.94)	26.3 (4.32)	27,307.78 (42.627)	45.5 (2.82)
CCM-2	88,382.19 (37,049.04)	36.9 (3.15)	27,186.13	80.1 (4.33)
CCM-3	48,788.88 (27,513.52)	92.7 (30.27)	31,075.12 (904.550)	143.0 (32.45)

Method	**Heuristic followed by strict**		**Strict followed by heuristic**	
	$\langle z \rangle^{10}$	$\langle t \rangle^{10}$	$\langle z \rangle^{10}$	$\langle t \rangle^{10}$
CCM-1	27,295.11 (14.296)	49.8 (3.14)	27,301.39 (28.331)	54.2 (2.10)
CCM-2	27,186.13	73.0 (3.29)	27,186.13	99.5 (5.21)
CCM-3	27,323.06 (159.377)	225.7 (14.49)	28,195.18 (875.116)	239.1 (48.56)

the crewblock construction method. Further, the average objective value is reduced significantly if the heuristic dominance rule is turned on after the strict dominance rule, given that crewblocks are constructed with CCM-3. We fail to reject the null hypothesis of equal sample means if we apply the test on the results obtained with both combinations. For example, the p-value of the t-test is 0.089 if CCM-3 is used. The null hypothesis of equal sample means can not be rejected at a significance level of 5%.

The Student's t-test is also used to evaluate whether or not the average computation time is significantly different if another (combination of) dominance rule(s) is used. It turns out that the running time is significantly larger if the strict dominance rule is turned on instead of the heuristic dominance rule. This reflects the characteristics of both rules. Furthermore, the average computation time is significantly lower if the strict dominance rule is used after the heuristic dominance rule instead of the reverse order. Promising results are shown if the strict dominance rule is turned on after the heuristic dominance rule.

In fact, the results between the heuristic and strict dominance rule are hard to compare, since the algorithm can terminate before no variable with negative reduced cost is identified in the pricing problem. Therefore, we also consider the case in which the parameters k and m are set to infinity. Again, ten independent runs are performed. Given that the heuristic dominance is turned on, the algorithm terminates on average after 54.6 seconds if CCM-1 is used, 76.4 seconds if CCM-2 is used and 162.1 seconds if CCM-3 is used. The average objective values are 39,203.31, 30,028.49 and 31,017.33, respectively. Based on the results stated in Table 20, we can conclude that other (combinations of) dominance rule(s) show more promising results.

Recall that the optimal solution of the master problem provides a lower bound on the optimal integer solution. The objective value of the optimal LP solution is $26,869.78$ if CCM-1 is used, $26,737.65$ if CCM-2 is used and $26,652.44$ if CCM-3 is used. It takes roughly 47 seconds to find the optimal LP solution if CCM-1 is used, 80 seconds if CCM-2 is used and 507 seconds if CCM-3 is used. These are the average computation times over five independent runs, given that the strict dominance rule is turned on. The solutions obtained for the original program are the same for all these independent runs. The objective value of the original problem is 27,288.72 if CCM-1 is used, 27,186.13 if CCM-2 is used, and 27,125.55 if CCM-3 is used. The upper bounds on the optimality gaps are given by 1.6%, 1.7% and 1.8%, respectively.

Consider the results obtained when the strict dominance rule is turned on. The upper bound on the optimality gap is on average approximately 1.6% if CCM-1 is used, 1.7% if CCM-2 is used, and 16.6% if CCM-3 is used. Consider the results obtained if the strict dominance rule is applied after the heuristic dominance rule. In that case, the upper bound on the optimality gap is on average approximately 1.6% if CCM-1 is used, 1.7% if CCM-2 is used, and 2.5% if CCM-3 is used.

The result for problem instance 5C are given in Table 21. The algorithm is implemented such that independent pricing subproblems are solved in parallel. Further, k is 20 and m is 50. The MIP of the RMP is solved after each iteration.

Table 21: The average result for problem instance 5C over ten independent runs for different (combinations of) dominance rules.

Method	**Heuristic dominance rule**		**Strict dominance rule**	
	$\langle z\rangle^{10}$	$\langle t\rangle^{10}$	$\langle z\rangle^{10}$	$\langle t\rangle^{10}$
CCM-1	1,261,975.00 (114,460.413)	41.2 (6.14)	31,258.05	321.3 (17.23)
CCM-2	1,055,505.12 (111,073.893)	52.9 (3.23)	30,191.30	537.1 (34.02)
CCM-3	728,717.90 (40,722.782)	135.5 (13.48)	30,859.88 (827.516)	2898.8 (45.82)
Method	**Heuristic followed by strict**		**Strict followed by heuristic**	
	$\langle z\rangle^{10}$	$\langle t\rangle^{10}$	$\langle z\rangle^{10}$	$\langle t\rangle^{10}$
CCM-1	31,299.65 (93.028)	302.1 (23.45)	31,258.05	320.8 (13.28)
CCM-2	30,836.34 (590.177)	456.2 (52.67)	30,196.82 (12.336)	537.4 (25.02)
CCM-3	31,574.42 (142.002)	2252.4 (130.89)	31,359.88 (48.561)	2897.5 (318.33)

Again, the solution quality is significantly worse if the heuristic dominance rule is used instead of the strict dominance rule or a combination of dominance rules. We can not reject the null hypothesis of equal sample means if the first sample consists of the objective values obtained when the strict dominance rule is used and the second sample consists of the objective values obtained when the strict dominance rule is turned on after the heuristic dominance rule, irrespective of the crewblock construction method. It takes roughly 2 seconds to solve the pricing problem if the heuristic dominance rule is turned on, and 26 seconds if the strict dominance rule is turned on.

Overall, the most promising results are obtained by using the strict dominance rule or by using the combination of heuristic dominance rule followed by the strict dominance rule.

7.2.3 Effect of a split per day of the week

We have shown in Subsection 7.1.2 that the solution time is reduced significantly if the pricing problems are split per day of the week and solved in parallel for the second data instance. In this subsection, we evaluate the effect of splitting the pricing problems per day of the week on the solution time and the corresponding computation time for larger datasets.

The algorithm is implemented such that the pricing subproblems are solved in parallel. The strict dominance rule is turned on. Further, k is 20 and m is 50. The MIP formulation of the RMP is solved after each iteration. The results are shown in Table 22. The total number of shifts constructed is given by '# shifts'.

Table 22: The average results over ten independent runs obtained when the pricing subproblems are solved over the entire network and when the pricing subproblems are split on weekday. Results are given for problem instances 5A, 5C and 6C

Dataset	**Method**	z^*_{LP}	**Pricing problems solved for the entire week**			**Pricing problems solved per day of the week**		
			$\langle z \rangle^{10}$	$\langle t \rangle^{10}$	**# shifts**	$\langle z \rangle^{10}$	$\langle t \rangle^{10}$	**# shifts**
5A	CCM-1	26,869.78	27,307.78 (46.627)	45.5 (2.82)	75.6 (1.86)	27,288.72	13.8 (0.12)	77.2 (2.59)
	CCM-2	26,737.65	27,186.13	80.1 (4.33)	106.6 (3.38)	27,186.13	21.3 (3.15)	108.0 (3.39)
	CCM-3	26,652.44	31,075.12 (904.550)	143.0 (32.45)	171.8 (28.99)	27,255.65	57.3 (4.32)	247.0 (11.55)
5C	CCM-1	30,376.99	31,258.05	321.3 (17.23)	94.8 (1.03)	31,258.05	103.9 (5.41)	93.2 (1.25)
	CCM-2	30,002.68	30,191.30	537.1 (34.02)	126.4 (2.97)	30,191.30	157.8 (3.12)	130.8 (2.18)
	CCM-3	29,891,31	30,859.88 (827.516)	2898.8 (45.82)	271.4 (10.46)	30,138.62 (35.620)	399.0 (31.54)	350.6 (23.86)
6C	CCM-1	Not solved	56,101.40	1606.2 (54.21)	189.2 (4.30)	56,101.40	588.7 (36.18)	194.1 (5.06)
	CCM-2	Not solved	53,488.89 (48.327)	2974.1 (331.79)	331.6 (8.34)	53,452.30 (16.96)	1276.3 (56.227)	343.5 (9.47)
	CCM-3	Not solved	53,296.41 (73.669)	23,658.3 (733.90)	893.2 (34.38)	53,318.20 (104.380)	7667.2 (461.95)	892.1 (41.26)

The Student's t-test is used to evaluate whether the average computation time is reduced significantly if the pricing problems are split per day of the week. We must reject the null hypothesis of equal sample means, irrespective of the dataset and crewblock construction method. This means that the average computation time is reduced significantly if the pricing problems solved per day of the week. Besides that, the average objective value at the moment of termination does not differ significantly or is significantly smaller if the pricing problems are split per day.

Significant performance improvements are obtained by splitting the pricing problem into multiple, smaller pricing subproblems. The pricing subproblems are constructed such that all resource feasible paths in the pricing problem solved over the entire network can still be constructed. The individual pricing subproblems are easier and faster to solve if we split on weekday. This is verified by the fact that the number of shifts constructed per second is on average larger if pricing subproblems are solved per day of the week.

7.2.4 Effect of the number of crewblocks

In this subsection, we evaluate the effect of the number of crewblocks on the solution time and the quality of the solution. Given a subset of crew requirements, we expect that the optimal MIP of the master problem is the lowest when every crewblock consists of exactly one workblock. On the other hand, the problem is easier and faster to solve if multiple workblocks are merged into one crewblock.

Consider the results for problem instances 5A and 5C presented in Table 22, where the pricing problem is split by day of the week. Recall that $|C|$ is equal to 71 if CCM-1 is used, 83 if CCM-2 is used and 141 if CCM-3 is used, considering problem instance 5. We can conclude from this table that the optimal solution of the master problem decreases with the number of crewblocks in C. Furthermore, the average computation time before termination grows with the number of crewblocks. The number of paths that must be researched in each pricing subproblem grows with the number of crewblocks included in the graph. The Student's t-test is used to evaluate whether or not the computation time is reduced significantly if less crewblocks are added to the formulation. We must reject the null hypothesis of equal sample means for all t-tests that we perform on the runtimes. Hence, the runtime is reduced significantly if CCM-1 is used instead of CCM-2, and if CCM-2 is used instead of CCM-3. The quality loss is only marginally, considering the optimal LP solutions. As a result, it is computationally attractive to merge multiple workblocks into one crewblock.

7.2.5 Effect of the data size and complexity of the break requirements

First, we evaluate the effect of an increase in the number of crew requirements on the solution time. The number of paths in each pricing subproblem grows exponentially with the number of crew requirements. Based on this, we expect that the computation time also increases exponentially with the number of crew requirements. Next, the effect of the break requirements on the solution time is investigated.

Table 23 shows the average runtime for different problem instances. The computation time is limited to 12 hours. Crewblock construction method CCM-1 is used, and the strict dominance rule is turned on. The pricing problems are split by weekday and solved in parallel. The MIP is solved after every five iterations.

Table 23: Average computation time (in seconds) before termination. The standard deviations are listed between brackets

Problem instance	**5**	**6**	**7**	**8**
A	13.8 (0.12) [1]	66.3 (2.01) [1]	713.9 (8.76) [2]	4108.5 (157.68) [2]
B	31.1 (3.19) [1]	169.5 (3.45) [1]	1060.8 (54.83) [2]	21,337.8 (765.22) [3]
C	103.9 (5.41) [1]	588.7 (36.18) [1]	6218.0 (213.41) [2]	$>$ 43,200.0

[1] Average computation time over 10 independent runs, [2] Average computation time over 5 independent runs, [3] Average computation time over 2 independent runs

The average computation time grows exponentially with the number of crew requirements. It takes roughly 31 seconds to solve problem instance 5B, while solving problem 8B takes roughly 6 hours. The only difference between these two problem instances is the number of crew requirements that must be covered. The number of nodes in each pricing subproblem grows with the number of crewblocks and passenger activities that can be included in the shifts generated inside the subproblem. The number of arcs grows more drastically than the number of nodes. As a result, the number of (feasible) paths grows exponentially with the number of train activities planned in the model week.

Table 23 shows that the average computation time is significantly higher if break requirements are introduced. The complexity of these break requirements also have a significant impact on the solution time. In Chapter 8 we describe a solution method that can be used to reduce the computation time if a meal break is required in the shifts.

The algorithm is not able to solve the CSP of dataset 8C within 12 hours. Most real-life problem instances are much larger than this instance, both in the number of employment groups and the number of train activities scheduled in the model week. Based on this, we conclude that the column generation technique described in this paper is not able to solve real-life CSPs within a reasonable time limit.

7.2.6 Effect of the number of employment groups

In this subsection, we investigate the effect of the number of employment groups on the solution time and solution quality. Recall that we have assumed that each employment group has the route knowledge, traction knowledge and skills to perform all crew requirements.

Results of computation experiments that can be used to evaluate the effect of the number of employment groups on the solution quality and computation time are given in Table 24. The pricing problem is split by employment group, nationality, break requirement and day of the week. The pricing subproblems are solved in parallel, with the number of threads depending on the number of CPU cores. The strict dominance rule is turned on. Furthermore, k = 20, m = 50 and the MIP formulation of the RMP is solved every five iterations. Crewblock construction method CCM-1 is used.

Table 24: Results of computational experiments to evaluate the effect of the number of employment groups on the solution quality and computation time

Dataset	$\langle z \rangle$	$\langle t \rangle$	$\langle Y_d \rangle$	$\lvert Q \rvert$	Average running time per iteration (s)
5C	31,258.05[1]	103.9 (5.41)[1]	12[1]	21	10.2 (3.25)[1]
5E	27,629.18[2]	11,547.3 (431.87)[2]	11[2]	210	722.5 (58.31)[2]
6C	56,101.40[1]	588.7 (36.38)[1]	21[1]	21	25.3 (4.98)[1]
6E	44,553.13 (233.14)[2]	24,752.1 (391.77)[2]	18[2]	210	556.8 (806.44)[2]

[1] Average over 10 independent runs, [2] Average over 5 independent runs

Table 24 shows that the average objective value at the moment of termination is lower when there are ten employment groups instead of one. Better shifts can be constructed, such that the crew deadheading duration and idle time in the shifts is reduced. Furthermore, the number of employees needed to cover the selected shifts is smaller when the number of employment groups is ten instead of one. The results show that the average computation time is significantly larger if shifts are constructed for ten employment groups. In fact, there are at least two reasons for this. First, the number of pricing subproblems that is solved in each iteration of the solution process is 21 if there is only one employment group and 210 if there are ten employment groups. Second, it is likely that the number of shifts generated in each iteration of the solution process is larger if 210 pricing subproblems are solved instead of 21. More shifts are added to the formulation and it takes significantly more time to solve the MIP of the RMP to optimality.

8 Conclusion and Further research

In this thesis, a large real-life problem instance of the crew scheduling problem (CSP) has been considered. The CSP is one of the most challenging planning problems faced by railway companies, and the problem is proven to be NP-complete. In the past, crew schedules were constructed with heuristic methods that were based on the rules used by manual planners. The planning puzzle became more complex over the last three decades, leading to an increasing demand for computer-assisted approaches and decision support systems that can construct good quality crew roster schemes automatically. In order to create a satisfactory crew schedule for the employees and management of the railway company, several complex rules and constraints need to be addressed. For example, the crew schedule must fulfill the government regulation concerning shift time and break requirements, and labour agreements. Furthermore, the traction knowledge, route knowledge and skills of the crew members must be considered. Existing literature is skewed towards mathematical programming and metaheuristic approaches.

The CSP is formulated as a set covering problem (SCP) with side-constraints. Every column of the formulation is a shift that can be carried out by one employment group, and that addresses the labour rules and government regulations. The side-constraints ensure that the set of selected shifts fulfills the comprehensive constraints, e.g., the percentage of overnight stays, regulating crew capacity, etc. First, a large candidate list of feasible shifts is constructed, given the set of crew requirements that must be covered. Second, the minimum covering set of shifts is selected from the candidate list by using the set covering formulation. Unfortunately, the explicit construction of all variables is computationally intractable. Column generation techniques and metaheuristics are examples of solution methods that work well in practice.

In this thesis, we proposed a column generation method to solve the CSP. Workblocks are constructed in the first stage of the process. A workblock is the sequence of crew requirements between two relief opportunities on a locline, i.e., the smallest amount of work that must be performed by one driver. We can reduce the problem size by merging multiple workblocks into one crewblock, such that the problem is easier and faster to solve. There is a trade-off between the number of workblocks merged and the solution quality. In this thesis, we proposed three crewblock construction methods. The first method starts with crewblocks that consist of all crew requirements planned in sequence on a mobilized locomotive. Thereafter, the method splits the crewblocks based on cutting rules. Each workblock is included completely in one crewblock. The second method is based on block cutting heuristics and constructs all possible crewblocks, given some predefined parameters. First, a large candidate list of crewblocks is constructed, given a set of cutting rules. The problem to select a subset of crewblocks such that every workblock is included in exactly one of the selected crewblock is formulated as a set partitioning problem. The third method merges workblocks based on predefined parameters, such as the minimum and maximum length of a crewblock. This method is extended with a local search heuristic to improve the initial solution. Next, the hand-overs of responsibility between the drivers involved in crew changes on a mobilized locomotive are scheduled.

The problem to construct a feasible crew schedule is decomposed into two subproblems, namely the restricted master problem (RMP) and the pricing problem. The RMP is the linear programming (LP) relaxation of the set covering formulation in which only a subset of variables is considered. Shifts that have the potential to improve the solution of the RMP are identified in the pricing problem. The pricing problem is formulated as a resource constrained shortest path (RCSP) algorithm. Resources are added to the labels to record, among others, the shift duration, break duration, night time spend and nationality. We must split the pricing problem by employment group and break requirement, to ensure that the shifts generated inside the pricing problem obey the naturally occurring constraints. The RCSP uses the concept of dominance rules to limit the number of paths that must be researched. We have considered the heuristic dominance rule and the strict dominance rule. We also investigated the option to split the resulting pricing subproblems by day of the week.

Small problem instances with up to 120 train activities planned in the model week were created for which the optimal solution to crew scheduling is known. The column generation method was terminated if no variables with negative reduced cost were identified in the pricing problem. If the strict dominance rule is turned on, all or almost all paths must be considered to find the resource feasible path with minimal reduced cost. In the end, the optimal solution of the LP relaxation of the original program is found. Significantly fewer paths are considered if the heuristic dominance rule is turned on, and promising shifts are identified that are not necessary optimal. Unfortunately, the algorithm can terminate before the optimal solution of the LP relaxation of the original program is found.

For the first problem instance, containing 84 crew requirements and two employment groups, the average optimality gap over ten independent runs is 14.5% if the heuristic dominance rule is turned on, and 10.4% if the strict dominance rule is turned on. Further, we could not reject the null hypothesis of equal run times for the two dominance rules. The algorithm is able to reach the optimal LP solution within 18 seconds. For the second problem instance, containing 120 crew requirements and one employment group, we concluded that significant performance improvements are obtained by splitting the pricing problem into multiple, smaller pricing subproblems. The pricing subproblems are easier and faster to solve. Moreover, multiple pricing subproblems can be solved in parallel. The number of threads depends on the number of CPU cores available. The computation time decreases significantly if the MIP of the RMP is only solved in the last iteration of the algorithm. For the third problem instance, containing 112 crew requirements and one employment group, the average optimality gap over ten independent runs is 191.6% if the heuristic dominance rule is turned on. The average computation time is 35 seconds. The optimal solution to crew scheduling is attained in all ten independent runs if the strict dominance rule is turned on, within approximately 126 seconds. For the fourth problem instance, the results obtained with both dominance rules are quite similar. Overall, the strict dominance rule outperforms the heuristic dominance rule with respect to the solution quality.

In addition to the small datasets, different problem instances are derived from the real-life problem instance to investigate the robustness of the method to different parameter settings, implementation options and data sizes. These datasets contain 350 to 2464 crew requirements. Several instances are constructed for each subset of crew requirements. One instance is the basic instance, and then we either add a break (in two different ways) and consider one or ten employment groups.

We have investigated the effect of the strict and heuristic dominance rule on the performance of the method. Furthermore, we investigated the effect of combinations of dominance rules on the solution quality. The strict dominance rule and the combination of heuristic dominance followed by strict dominance show the most promising results.

Significant performance improvements are obtained by splitting the pricing problem into multiple, smaller pricing problems. The pricing subproblems are easier and faster to solve if we split on day of the week. Furthermore, we have shown that it is computationally attractive to merge multiple workblocks into one crewblock. In that case, the runtime is reduced significantly, while the objective value increases only marginally.

Recall that the CSP is NP-complete. We found that the computation time increases exponentially with the number of crew requirements that must be assigned. Furthermore, the computation time increases with the complexity of the break rules, as the pricing subproblems are solved over larger networks. The runtime increases with the number of employment groups included in the problem, since more pricing subproblems must be solved in that case. On the other hand, the cost of the crew schedule decreases with the number of employment groups. More effective shifts can be constructed in that case.

We will provide some ideas for further research in the remainder of this chapter.

In this thesis, we proposed different methods to merge multiple workblocks into one crewblock, such that the size of the CSP is reduced. Even though the results were satisfactory, there is room for improvement in constructing the crewblocks. For example, it is worth investigating whether performance improvements are realized if the number of workblocks is reduced in the first step of the solution process. We assumed that a crew change on a mobilized locomotive can be planned if the idle time between two train activities is at least equal to the hand-over duration of one minute, and if the corresponding location is a relief location. In practice, crew changes are only planned if the idle time between two crew requirements is, for example, more than 15 minutes. Furthermore, the problem size is reduced if multiple crewblocks, scheduled on one or multiple loclines, are already combined before the column generation algorithm. For example, crewblocks are combined based on their combination quality. The quality of a crewblock combination can be evaluated in terms of the idle time between the crewblocks, and the crew requirements covered.

A crucial element in the CSP is the set of complex break requirements that are applicable to the crew. Given that the first break must start in some predefined time window after the start of the shift, multiple nodes are required for each time point and location at which at least one event starts and/or ends. The resource windows associated with these nodes are different, to ensure that the breaks are planned according to the break rules. However, it is sufficient to construct one node for each time point and location at which at least one event starts and/or ends if no break window is set. A break is scheduled on an arc if the break time is larger than or equal to the minimum break duration, irrespective of the spread time of the shift. Hence, the number of paths that must be researched in the pricing subproblems is significantly smaller. When a break window is specified, it might be computationally attractive to initially omit this window. Shifts are constructed with the column generation algorithm until a stopping criterion is satisfied. Then, we must check whether or not the selected shifts satisfy the break requirements. The crew requirements included in the shifts that do not satisfy the break rules must be re-scheduled. If only a small number of crew requirements must be re-scheduled, manual planning or heuristic methods can be used. Otherwise, the column generation algorithm is used again. The time window in which the first break should start must be considered during re-scheduling.

We have already mentioned that there is no need to solve each pricing subproblem to optimality in every iteration of the solution process. It is likely that the runtime decreases if only a limited number of subproblems is solved by turning the strict dominance rule on, and that the other subproblems are solved by using the heuristic dominance rule. Moreover, not all pricing subproblems have to be solved in each iteration of the solution process. However, we can only conclude that the optimal solution of the master problem is found if all pricing subproblems are solved to optimality and no variable with negative reduced cost is identified.

Even for the small problem instances, the runtime decreased significantly if the MIP of the RMP was not solved in each iteration of the solution process. It might be interesting to investigate stopping criteria that do not require the objective value of the MIP formulation. For example, both k and m depend on the LP solution rather than the MIP solution. In that case, it suffices to solve one MIP of the RMP after the column generation process is terminated.

The number of arcs in the graphs on which the pricing subproblems are stated turned out to be a point of interest, especially for large problem instances. It is theoretically possible to construct wait, crew deadheading, arrival and departure arcs at any point in time. In that case, the number of arcs in the graphs would become extremely large, which results in memory issues and makes the problem very hard to solve. Therefore, we already omitted a very large number of these arcs. However, there is room for improvement in generating these arcs. We will describe some of these options in the following four paragraphs.

First of all, the number of wait arcs with a break might be reduced. For example, only one wait arc with break is constructed in each node, with a break larger than or equal to the minimum break duration. Even though the solution space decreases by doing so, we do not expect large issues.

Secondly, suppose that a driver waits a location before another activity is planned in his shift. It is likely that this activity corresponds to a crewblock or passenger activity that starts at the current location, or that it involves a crew deadheading operation. In the last case, we could also directly include a crew deadheading arc in the shift, instead of a wait arc followed by the same deadheading operation. In terms of the graph, it might be beneficial to construct wait arcs and crew deadheading arcs with a head node corresponding with the start of a crewblock or passenger activity only. In that case, the number of paths that must be researched in each pricing subproblem is reduced. For example, suppose that a wait arc is constructed between node v and node w, where w does not correspond with the start of an event. It is likely that a wait arc or crew deadheading arc with a tail node in w is included in the path after arc (v, w). Suppose that crew deadheading arc (w, u) is included. It was also possible to use the deadheading arc (v, u).

Thirdly, the number of departure and arrival arcs can be reduced as well. In practice, it is sufficient to construct departure arcs only between a crew depot and a node that corresponds with the start of an event. Furthermore, arrival arcs should only be constructed between nodes that correspond with the end of an event and the crew depots.

Finally, we have constructed departure, arrival and crew deadheading arcs between two nodes if the deadhead time does not exceed the maximum spread time of the shift. However, long deadhead arcs are very ineffective and would hardly be used. It might be beneficial to limit the deadhead duration on an arc, thus omitting some of the arcs that were constructed before.

Each driving activity planned in the model week can be used to transfer drivers as a passenger on a train. We have constructed two nodes for each driving operation, corresponding with the start and end of the event. The passenger activity is represented by the arc between these nodes. The number of nodes corresponding with passenger activities is large, and the size of the problem is reduced if some or all possible passenger activities are omitted. For example, we only include those passenger activities corresponding with a pair of locations between which no faster (combination of) deadheading options exists. When the crew schedule is constructed, the planner can try to reduce the cost by considering the option to travel as a passenger on a train.

For simplity, we have assumed that each trip with the expensive taxicab takes 3 hours between locations in the same country, and 7 hours otherwise. The euclidean distance can be used to estimate the actual driving times, given the coordinates of the locations. The cost associated with a trip with the expensive taxicab must depend on the duration of the ride.

The idea of dividing the problem into smaller subproblems which are solved in parallel seems to be rather efficient. The pricing problem must be split by employment group, nationality and break requirements to ensure that the generated variables satisfy the operational constraints. Furthermore, the performance improvement turns out to be significant if pricing subproblems are solved by weekday. It might be interesting to investigate other decomposition criteria, concerning geographical partitioning. For example, we can decompose the overall problem into overlapping subregions. Each crewblock and employment group is assigned to one primary subregion and possibly multiple secondary subregions, such that there is a balance between the solution quality and the solution time. The subregions are updated after several iterations.

The column generation method can be extended with column fixing techniques and a Lagrangian-based heuristic to speed up the solution process.

References

Abbink, E., Albino, L., Dollevoet, T., Huisman, D., Roussado, J., and Saldanha, R. L. (2011). Solving large scale crew scheduling problems in practice. *Public Transport*, 3(2):149–164.

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

Abbink, E., van 't Wout, J., and Huisman, D. (2008). Solving large scale crew scheduling problems by using iterative partitioning. Technical report, Econometric Institute Research Papers.

Bach, L., Dollevoet, T., and Huisman, D. (2014). Integrating timetabling and crew scheduling at a freight railway operator. Technical report, Econometric Institute Research Papers.

Ball, M. and Benoit-Thompson, H. (1988). A lagrangian relaxation based heuristic for the urban transit crew scheduling problem. *Computer-Aided Transit Scheduling, Lecture Notes in Economics and Mathematical Systems*, 308:54–67.

Banihashemi, M. and Haghani, A. (2001). A new model for the mass transit crew scheduling problem. In *Computer-Aided Scheduling of Public Transport*, pages 1–15. Springer.

Beasley, J. E. and Chu, P. C. (1996). A genetic algorithm for the set covering problem. *European Journal of Operational Research*, 94(2):392–404.

Blum, C. and Roli, A. (2003). Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Computing Surveys (CSUR)*, 35(3):268–308.

Caprara, A., Fischetti, M., and Toth, P. (1999). A heuristic method for the set covering problem. *Operations Research*, 47(5):730–743.

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

Cavique, L., Rego, C., and Themido, I. (1999). New heuristic algorithms for the crew scheduling problem. In *Meta-Heuristics*, pages 37–47. Springer.

Dantzig, G. B. and Wolfe, P. (1960). Decomposition principle for linear programs. *Operations Research*, 8(1):101–111.

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

Emden-Weinert, T. and Proksch, M. (1999). Best practice simulated annealing for the airline crew scheduling problem. *Journal of Heuristics*, 5(4):419–436.

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

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

Fischetti, M., Martello, S., and Toth, P. (1989). The fixed job schedule problem with working-time constraints. *Operations Research*, 37(3):395–403.

Fisher, M. L. (2004). The lagrangian relaxation method for solving integer programming problems. *Management Science*, 50(12_supplement):1861–1871.

Freling, R., Lentink, R. M., and Odijk, M. A. (2001). Scheduling train crews: A case study for the Dutch railways. *Computer-Aided Transit Scheduling, Lecture Notes in Economics and Mathematical Systems*, 505:54–67.

Glover, F. (1989). Tabu search-part i. *ORSA Journal on computing*, 1(3):190–206.

Glover, F. (1990). Tabu search-part ii. *ORSA Journal on computing*, 2(1):4–32.

Guo, Y., Mellouli, T., Suhl, L., and Thiel, M. P. (2006). A partially integrated airline crew scheduling approach with time-dependent crew capacities and multiple home bases. *European Journal of Operational Research*, 171(3):1169–1181.

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

Huisman, D. (2007). A column generation approach for the rail crew re-scheduling problem. *European Journal of Operational Research*, 180(1):163–173.

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

Jütte, S., Albers, M., Thonemann, U. W., and Haase, K. (2011). Optimizing railway crew scheduling at DB Schenker. *Interfaces*, 41(2):109–122.

Jütte, S. and Thonemann, U. W. (2012). Divide-and-price: A decomposition algorithm for solving large railway crew scheduling problems. *European Journal of Operational Research*, 219(2):214–223.

Jütte, S. and Thonemann, U. W. (2015). A graph partitioning strategy for solving large-scale crew scheduling problems. *OR Spectrum*, 37(1):137–170.

Laplagne, I. E. (2008). *Train driver scheduling with windows of relief opportunities*. PhD thesis, University of Leeds.

Qiao, W., Hamedi, M., and Haghani, A. (2010). Algorithm for crew-scheduling problem with bin-packing features. *Transportation Research Record: Journal of the Transportation Research Board*, 2197(1):80–88.

Ruxton, G. D. (2006). The unequal variance t-test is an underused alternative to Student's t-test and the Mann–Whitney U test. *Behavioral Ecology*, 17(4):688–690.

Savelsbergh, M. (1997). A branch-and-price algorithm for the generalized assignment problem. *Operations Research*, 45(6):831–841.

Sellmann, M., Zervoudakis, K., Stamatopoulos, P., and Fahle, T. (2002). Crew assignment via constraint programming: integrating column generation and heuristic tree search. *Annals of Operations Research*, 115(1-4):207–225.

Smith, B. M. and Wren, A. (1988). A bus crew scheduling system using a set covering formulation. *Transportation Research Part A: General*, 22(2):97–108.

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

Souai, N. and Teghem, J. (2009). Genetic algorithm based approach for the integrated airline crew-pairing and rostering problem. *European Journal of Operational Research*, 199(3):674–683.

Vaidyanathan, B., Jha, K. C., and Ahuja, R. K. (2007). Multicommodity network flow approach to the railroad crew-scheduling problem. *IBM Journal of Research and Development*, 51(3.4):325–344.

Willers, W., Proll, L., and Wren, A. (1995). A dual strategy for solving the linear programming relaxation of a driver scheduling system. *Annals of Operations Research*, 58(7):519–531.

Woodburn, A. (2012). Intermodal rail freight activity in britain: Where has the growth come from? *Research in Transportation Business & Management*, 5:16–26.

Appendices

A Operational constraints for the real-life problem instance

Table A25: Minimum and maximum spread time

Spread time requirement	Regime	International shift	Minimum shift duration	Maximum shift duration
S_1	R_1	TRUE	05:00:00	10:00:00
S_2	R_1	FALSE	05:00:00	09:00:00
S_3	R_2	TRUE	05:00:00	13:00:00
S_4	R_2	FALSE	05:00:00	10:00:00

Table A26: Maximum driving time

Driving time requirement	Regime	Night shift	International shift	Maximum driving duration
D_1	R_1	TRUE	TRUE	08:00:00
D_2	R_1	TRUE	FALSE	1 day
D_3	R_1	FALSE	TRUE	09:00:00
D_4	R_1	FALSE	FALSE	1 day
D_5	R_2	TRUE	TRUE	08:00:00
D_6	R_2	TRUE	FALSE	08:00:00
D_7	R_2	FALSE	TRUE	09:00:00
D_8	R_2	FALSE	FALSE	09:00:00

Table A27: Break requirements

Break requirement	Regime	Employee role	Minimum shift duration	Maximum shift duration	First break after	First break before
B_1	R_1	Global driver	05:00:00	2 days	00:00:00	08:00:00
B_2	R_1	Global driver	05:00:00	08:00:00	03:00:00	06:00:00
B_3	R_1	Global driver	08:01:00	2 days	03:00:00	06:00:00
B_4	R_2	Global driver	05:00:00	2 days	00:00:00	08:00:00
B_5	R_2	Global driver	05:00:00	08:00:00	03:00:00	06:00:00
B_6	R_2	Global driver	08:01:00	2 days	03:00:00	06:00:00
B_7	R_2	Local driver	05:00:00	2 days	00:00:00	08:00:00
B_8	R_2	Local driver	05:00:00	08:00:00	03:00:00	06:00:00
B_9	R_2	Local driver	08:01:00	2 days	03:00:00	06:00:00

Break requirement	International shift	Break duration	Split allowed	Minimum break duration	Move as break	Break in loc
B_1	FALSE	00:20:00	FALSE	00:15:00	TRUE	TRUE
B_2	TRUE	00:30:00	TRUE	00:15:00	TRUE	TRUE
B_3	TRUE	00:45:00	TRUE	00:15:00	TRUE	TRUE
B_4	FALSE	00:15:00	FALSE	00:15:00	TRUE	TRUE
B_5	TRUE	00:30:00	TRUE	00:15:00	TRUE	TRUE
B_6	TRUE	00:45:00	TRUE	00:15:00	TRUE	TRUE
B_7	FALSE	00:15:00	FALSE	00:15:00	FALSE	FALSE
B_8	TRUE	00:30:00	TRUE	00:15:00	FALSE	TRUE
B_9	TRUE	00:45:00	TRUE	00:15:00	TRUE	TRUE

B Student's t-test

Different versions of the Student's t-test exists. The aim of this section is to provide general information about the unequal variance Student's t-test. More information about this statistical hypothesis test can be found in the paper written by Ruxton (2006).

The test statistic follows a Student's t-distribution if the null hypothesis of equal means is supported. A t-test can be applied to two independent samples that come from a normal distribution. Suppose that the expected sample means are given by μ_1 and μ_2, respectively. We test the null hypothesis $H_0 : \mu_1 = \mu_2$ against $H_A : \mu_1 \neq \mu_2$. The size of the first sample is N_1 and the size of the second sample is N_2. The test-statistic T is given by:

$$T = \frac{\bar{Y}_1 - \bar{Y}_2}{\sqrt{s_1^2/N_1 + s_2^2/N_2}}$$

where $\bar{Y}_1$ ($\bar{Y}_2$) is the sample mean of the first (second) sample and s_1^2 (s_2^2) is the sample variance of the first (second) sample. The significance level is given by α. The null hypothesis of equal means is rejected if $|T| > t_{1-\alpha/2}, \upsilon$, where $t_{1-\alpha/2}, \upsilon$ is the critical value of the t-distribution with υ degrees of freedom. The degrees of freedom is calculated using the following formula.

$$\upsilon = \frac{(s_1^2/N_1 + s_2^2/N_2)^2}{(s_1^2/N_1)^2/(N_1 - 1) + (s_2^2/N_2)^2/(N_2 - 1)}$$