UNIVERSITY OF TWENTE

MASTER THESIS

Reducing driver movements in railway rescheduling by optimising a multi-commodity network flow model with a variable neighbourhood search

Author:
N. KORNEGOOR

Supervisor:
D. GUERICKE
A. TRIVELLA
R. KRANENDONK

A thesis submitted for the study program
Industrial Engineering and Management.

March 13, 2024

Acknowledgements

This report is the result of my graduation project at DB Cargo Nederland located in Utrecht, which I executed as the final assignment for my master Industrial Engineering & Management. This program taught me tools and skills applicable to an endless variety of processes and industries. During my studies at the University of Twente, I was able to develop myself both personally and professionally. My Bachelor of Industrial Design Engineering provided me with a valuable set of skills and perspective to approach varying problems. The choice for the Master Industrial Engineering & Management has been an excellent decision. As a teaching assistant, I enjoyed helping other students with several courses. I also had the opportunity to study abroad for a semester at the National Chiao Tung University. I look back at a wonderful time at the University of Twente.

I would like to thank DB Cargo for allowing me to perform my graduation assignment there. Thanks Reinier and Hedwig for the supervision and the opportunity to experience the different departments of the company. Joining a train driver during a shift is a unique experience that would otherwise not have been possible.

Furthermore, I would like to thank my supervisor, Daniela, for her guidance throughout this project. The rapidness of your e-mail responses, even outside of normal working hours, made me feel like I was your highest priority. Thanks, Alessio, for offering your view on this thesis, allowing me to make significant improvements.

Thanks to my parents for their unconditional support before, during and after this graduation process. And lastly, thanks to my sister for proofreading my thesis.

Nico Kornegoor *Enschede, March 2024*

Management Summary

Problem Definition

DB Cargo NL is a rail freight operating company located in Utrecht, the Netherlands. They noticed a disparity between realised taxi costs and the budget for taxi rides. Taxis are one of the methods used by DB Cargo to move train drivers between locations. To reduce taxi costs, DB Cargo acquired company cars, which could be planned instead of taxi rides. Taxi rides are still a large contributor to DB Cargos operational costs because of their high cost. Therefore, DB Cargo started this research to see if taxi usage can be reduced in the later planning stages.

From the analysis of the current situation, it is determined that many changes are made to the schedules between the Week Plan and the Day Plan. The Week Plan is the plan of the upcoming week, released a week before execution. The Day Plan is the plan for the upcoming day, which is released a day before the execution. Between these planning phases, the time spent moving drivers increases, while the time spent driving trains does not change. Due to the changes made in the period between Week Plan and Day Plan, the Day Plan has become less efficient. The core problem is defined as *There is no optimisation tool to optimise the Day Plan*. By developing an optimisation tool for the Day Plan, driver movements can be reduced in this planning phase. By reducing the total time spent on driver movements and smart scheduling of company cars, costly taxi rides can be avoided. The objective of the research is:

To develop an optimisation model that reschedules the Day Plan to reduce driver movements and optimise the use of company cars.

Solution Design

A three-step approach is used to achieve the research objective. The day plan is optimised by reducing the required driver movements. Following this, company car scheduling strategies are compared based on company car utilisation. Finally, the combined effect of optimizing the Day Plan and company car scheduling on taxi usage is determined. The rescheduling problem is represented on a Multi Commodity Flow Network, which allows the representation of crew and company car schedules on the same network.

A Variable Neighbourhood Search (VNS) is developed to minimise the required driver movements to perform all planned actions in the Day Plan. Experiments are conducted to compare different company car scheduling strategies. Optimised Day Plans are sequentially sampled to mimic the scheduling of multiple days. The scheduling strategies are compared to a baseline set by a First Come First Served strategy (FCFS). This baseline maximises short-term company car usage. It simulates what would happen if company cars were scheduled without regard to their

future planning. A scheduling strategy is developed that balances short- and long-term company car utilisation. Finally, taxi usage is evaluated using both original and optimised Day Plans with the FCFS and the best company car scheduling strategy.

Results & Findings

After optimisation with the VNS, the average driver movement time in the Day Plan was reduced by 11.47%. This reduction brings the driver movement percentage in the Day Plan to below the driver movement percentage in the Week Plan. The Week Plan has the lowest percentage of driver movement time of all planning phases. Figure 1 shows that long-term taxi usage decreases by ∼14% when the best found company car scheduling strategy is applied to optimised Day Plans. The reduction in driver movements results in an increase in driver idle time. During operations, idle drivers could be used to further reduce taxi costs by letting drivers relocate company cars to useful locations.

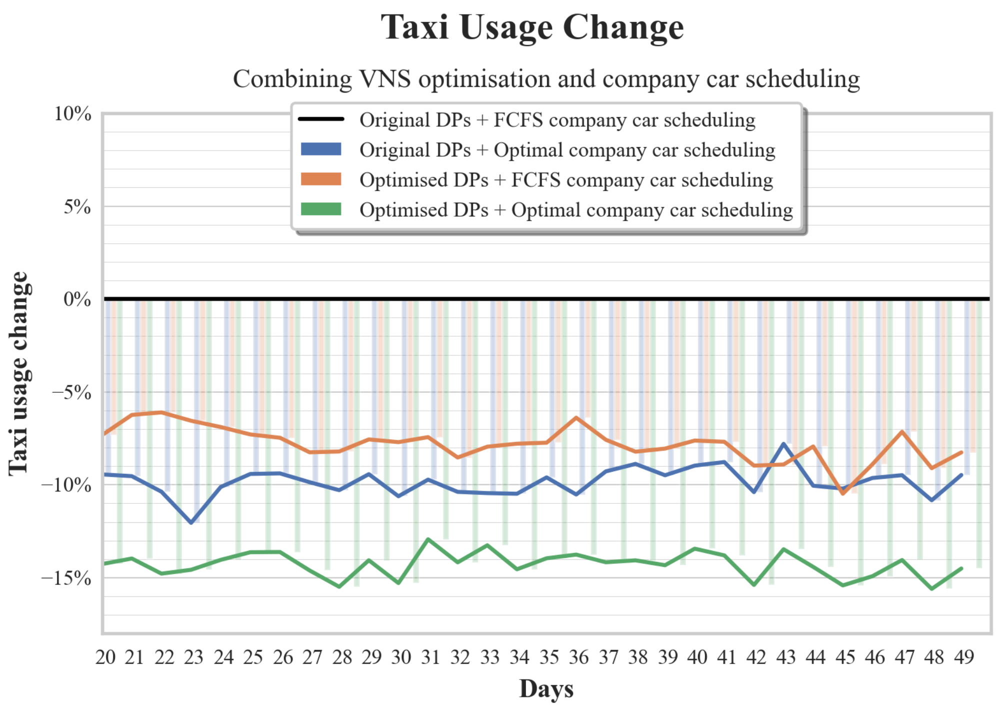

FIGURE 1: Taxi usage decrease with optimised Day Plans and company scheduling according to the best-found strategy in section 5.3

Practical Contribution

This thesis illustrates the potential for using optimisation models to improve the Day Plan's efficiency by reducing the required number of driver movements. Specific scheduling rules that are required for practical application are added to the rescheduling model. The experiments are of realistic size, and the model ensures that drivers end their shift at their start location. The model also respects shift end times to prevent problems with labour laws. Driver-specific material and route knowledge are excluded because the available data is anonymous. Rescheduling restrictions regarding route and material knowledge can be added to the model if the data is available.

The company car scheduling experiments indicate that it could be beneficial to add company cars to specific routes, as the company expected. It is not possible to provide specific route recommendations, as experiments have been conducted on newly generated data. The potential routes should be determined based on previous taxi usage. It is important to consider the number of locations that a restricted company car can travel to. A set that is too small decreases short-term company car usage, while a set that is too large decreases long-term performance.

Scientific Contribution

Railway crew rescheduling is a widely researched topic in the operations research field. The majority of railway crew rescheduling literature focuses on passenger railways, while less literature focuses on freight railways. Crew rescheduling often focuses on the reduction of overtime or minimising crew size. This research optimizes crew schedules based on driver movement time, which is, to the best of our knowledge, not regarded in literature before. The reduction of driver movement time holds particular significance for freight railway scheduling, as freight trains have greater irregularity than passenger trains. Freight trains also frequently depart from stations that regularly have no nearby passenger train stations. Due to these irregularities, taxis are a common method of moving freight train drivers. A new mathematical formulation is proposed based on a Multi-Commodity Network Flow to optimise crew and company cars simultaneously. While the VNS developed to optimise the model is unable to simultaneously optimise crew and company car schedules, it successfully optimises driver schedules by reducing driver movement time.

Contents

List of Figures

List of Tables

Acronyms

D&C	Destroy & Construct. 32, 33, 37, 43
DB	Deutsche Bahn. 2, 10
DBc	DB Cargo Nederland. x, 1–5, 7, 8, 13, 16, 19, 21–23, 38, 39, 54, 56
DP	Day Plan. vi, x, xi, 6–11, 16, 18, 19, 21, 29, 35–39, 41–43, 46–56, 62
F&C	Finance & Control. 3, 8
FCFS	First Come First Serve. 35–37, 46, 50–54, 63
GVNS	General Variable Neighbourhood Search. 31–33, 35, 37
KPI	Key Performance Indicator. 22
MCNF	Multi-Commodity Network Flow. 14, 15, 18, 19, 22–24, 26, 30, 31, 34, 56
RP	Resource Planning. 5, 6, 9
RSRP	Rolling Stock Rescheduling Problem. 14, 15, 18
RTM	Real Time Management. 6–10, 23, 54
STP	Short Term Planning. 5, 6
VND	Variable Neighbourhood Descent. 30–33, 37
VNS	Variable Neighbourhood Search. ix, 15, 18, 19, 30, 31, 43, 46, 48, 52–56, 69
WP	Week Plan. 3, 6, 9, 54, 55

1 Introduction

This thesis aims to reduce the use of taxis in the daily operations of DB Cargo Nederland (DBc). Costs are a major deciding factor in freight mode decisions (Cullinane & Toy, 2000). It is thus important to reduce operational costs in freight transport. Taxi rides are one of the methods used by DBc to relocate train drivers. Because taxi rides are the most expensive method to move drivers, DBc wishes to reduce the use of taxi rides. In this thesis, a model is developed that optimises the short-term plan by reducing driver movements. Based on optimised plans, company car scheduling strategies are evaluated and compared to a baseline strategy. These strategies aim to maximize company car usage. Together, the optimization of the short-term plan and the company car scheduling strategies should reduce the use of taxis.

The following chapter is an introduction that considers the context of the research. Section 1.1 provides background on the relevance of the research. Section 1.2 describes the context of the research, the company and the relevant departments. Section 1.3 analyses the current situation. From the analysis of the current situation, the problem is defined in section 1.4. The research objective is given in section 1.5 and the approach to solving the problem is given in 1.6.

1.1 Background

The railway freight sector has seen a significant increase in total transport volume between the '90s and '08. After 2008, this growth has slowed to a modest increase in total volume over the last couple of years (ACM, 2021). With the increasing demand for railway transportation, the rescheduling problem in railway networks has received more and more attention recently (Fang et al. 2015). Research in this area is scientifically challenging, but also promising from a practical perspective.

Compared to passenger rail transport, rail freight involves longer, less frequent trips with fewer opportunities for train driver exchanges. Exchanging trains is useful in driver scheduling, as it allows train drivers to be scheduled on a return trip. Because there are fewer opportunities for driver exchanges, it is harder to create schedules where drivers exchange trains without relocating the drivers. DBc resorts to taxis as one of the methods to move train drivers between locations. To reduce the costs of these taxi rides, they acquired company cars. These cars can be planned instead of using taxis and are driven by train drivers.

Most railway rescheduling research focuses on optimizing timetables, rolling stock, and crew schedules. The literature review in chapter 2 shows that while crew rescheduling literature aims to minimize crew-related costs like overtime, there's a lack of specific focus on minimizing driver movements or taxi costs. Such optimization can significantly benefit the rail freight sector, where driver movements without trains are more common. Taxi rides (and company cars) are often used as a method to move train drivers by DBc and its German counterpart, DB Cargo.

1.2 Context

This section describes the context in which the assignment takes place. It discusses the company, gives a broad overview of the railway scheduling process, and describes the given assignment. The next section (1.3) describes how the company's different departments perform the scheduling process.

1.2.1 The Company

DB Cargo Nederland (DBc) is the Netherlands' largest supplier of rail freight transport, with a current market share of 40.4% (DBC, 2023). DBc is the former freight division of NS. It was acquired by Deutsche Bahn in 2000 and is now a subsidiary of *DB Cargo AG*, Europe's largest freight railway undertaking, whose rail transport division has a market share of 18% (ACM, 2021).

1.2.2 Railway planning

Most railway companies' scheduling processes consist of sequentially planned timetables, rolling stock and crew planning, generally in that specific order (Cacchiani et al. 2014). Railway planning is typically a sequential problem, where each stage is planned relatively independently due to the size and complexity of the individual stages. A typical sequence of planning problems as described by Lusby et al. (2018) are:

Network Design The topological structure of the rail network; station locations and capacities.

Line Planning Subsets of stations and a route between them that constitute train lines.

Timetabling Determining exact times for events that should occur for rail units such as driving between stations.

Rolling Stock Planning The problem of allocating rolling stock units to the timetabled trips in circulations, considering compositions and depot parking.

Crew Rostering Creating rosters of duties to be performed by specific crew members during their shifts.

Rescheduling Solving problems before and during operations, managing delays and disruptions.

At DBc, a similar sequential approach to railway planning can be recognised. Figure 1.1 shows the sequential planning steps with related time-frames and DBc's departments that are involved in the planning step. Network Design is not included, as this is the network owner's responsibility. Section (1.3) describes how the departments perform parts of the scheduling process as well as rescheduling.

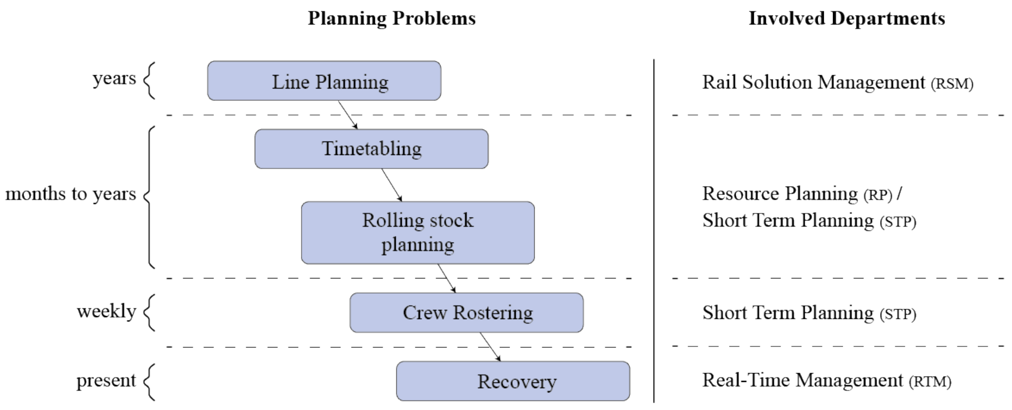

FIGURE 1.1: Planning Process of DBc. *Adjusted from:* Lusby et al. (2018)

1.2.3 The assignment

For several years, the Department of Finance & Control (F&C) noted a deviation between the realised costs of taxis used to move train drivers and the budget for these taxi rides. The taxi costs per planning phase and their relation to the budget are shown in figure 1.2. The planning phases are explained in section 1.3. F&C also states that the taxi costs are increasing yearly. Therefore, they would like to research how taxi costs can be reduced. In 2022, the realised taxi costs were much higher than expected from the long-term plan (Resource Plan in figure 1.2). The largest increase occurred between the Week Plan, which is the planning for the upcoming week, and the realisation. In this period, the schedule changes because of unexpected disruptions such as maintenance or infrastructure breakdowns. These disruptions require rescheduling of the original plan. In this rescheduling, many driver movements are added to the planning. In the end, the realised taxi costs are approximately 42% over budget (figure 1.2). Of these costs, only half occur in the long-term plan. Based on historical results, F&C believes there is enough budget for unexpected taxi rides and that some realised taxi costs are preventable. They expect that a decision support tool for rescheduling during operations can decrease taxi costs.

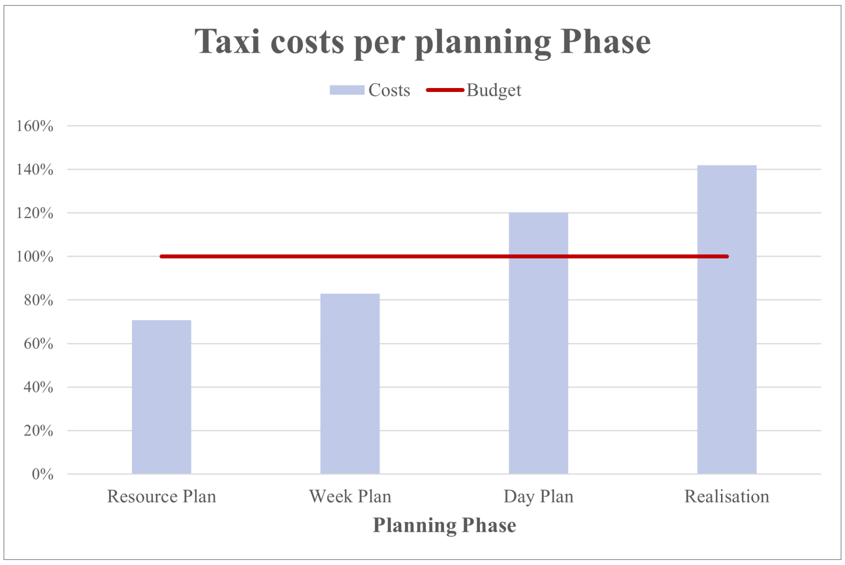

FIGURE 1.2: Taxi cost per planning phase

1.3 Current Situation

The typical railway planning sequence, as described in section 1.2.2, can be recognised at DBc. This section describes what a normal crew schedule looks like. It also describes the planning process per department, the rescheduling process, and how both planning and rescheduling add to taxi costs. Finally, it discusses how DBc uses company cars to decrease taxi costs.

1.3.1 Train driver shift

DBc's trains are driven by their train drivers. The train drivers have a schedule that determines the sequential actions they should undertake during their shifts. Crew schedules are made up of activities from the following categories:

- **Train rides** Driving a train. With or without carriages.
- **Productive actions** Actions that have to be performed related to trains. These can be starting or inspecting trains, but shunting and connecting carriages are also examples of these activities.
- **Movements** When drivers need to move to reach the start location of the next planned action, a movement is planned. Movements can be planned using passenger rail transport, company cars, taxi rides or even walking. Movements are also used to ensure that drivers end their shift at the start location.
- **Breaks** Every shift needs a 30-minute break.

Every train driver has a crew base with a related starting location. Every driver starts and ends their shift in their starting location. When a train trip is assigned to a crew schedule, the required movements of the train driver are also planned. These are all movements required to reach the next action and end the shift in the starting location. Crew schedules can also contain other productive actions related to a train. These often happen before or after a train ride, but a driver can also perform these tasks without driving the train itself. An example crew schedule is shown in table 1.1. The actions are categorized according to the categories given before.

Action	Description	Start time	End location	Category
DA	Company Car ride	Aml 8:00	Em 9:44	Movements
189	Train ride noted with the locomotive number	Em 9:44	Amf 11:21	Train rides
AR	Shunting activity	Amf 11:21	Amf 11:26	Productive actions
VR	Shunting activity	Amf 11:33	Amf 11:38	Productive actions
VP	Connection Test	Amf 11:38	Amf 11:48	Productive actions
193	Train ride noted with the locomotive number	Amf 11:48	Em 13:22	Train rides
SCHAFT	Break	Em 13:27	Em 13:57	Breaks
DA	Company Car ride	Em 14:02	Aml 15:32	Movements

TABLE 1.1: Example driver schedule.

1.3.2 Resource planning

The Resource Planning (RP) is the longest planning horizon of DBc. Most train rides in the RP are cyclic, occurring for instance once every day or week. Train rides are based on customer orders, which are typically long-term agreements. This can be months or years. After planning a train, the department also assigns the related driving action to a crew schedule. In the RP crew schedules are anonymous and only assigned to a crew base, not a specific driver.

When possible, train drivers ride trains as a return trip, as seen in the example schedule in table 1.1. Sometimes, drivers have to be relocated without driving a train. In the RP, passenger trains are the preferred method to move drivers. Not all places can be reached by passenger trains, and during night shifts, passenger trains are not an option. Company cars are a possible solution to these problems. Companies cars are often faster than passenger trains and can reach locations that are not near a train station. A company car has to be specifically planned to be at a location, which is a disadvantage. When a driver uses a company car, the car remains at that location afterwards. When company car trips are planned in the RP, these trips are also assigned to a specific company car. A last option to move drivers in the RP are taxis. Because of the cyclic nature of the RP planned taxi rides are very costly and are therefore avoided as much as possible. Taxis are used because company cars are limited and scheduling a company car is not always efficient.

Train trips are regularly uneven, meaning there are more outgoing trains (NL $\rightarrow$ EU), than incoming trains (EU $\rightarrow$ NL). Planning a company car to go to a location also requires a train driver to return the company car at another time. The unevenness of trains complicates planning return trips of the company cars. Taxis can be used to avoid having company cars idle at faraway locations. Furthermore, taxi rides are good for short trips because they don't require much planning. Because of the cyclic nature of the RP, taxi costs are already significant in this planning phase. $\sim$50% of the realised taxi costs exist in the resource plan. However, the amount of taxi costs is still within budget (at $\sim$70%). This is shown in figure 1.2.

Besides scheduling, the department also implements changes required due to long-term maintenance that is known far in advance. An example of such long-term maintenance is the upgrade of the Dutch-German border crossing. This required re-routing of trains over the Venlo border crossing for a long period.

1.3.3 Short-term planning

The Short Term Planning (STP) Department has mostly the same tasks as RP but for a period of 4 weeks to a day before the realisation. STP implements irregularities into the RP which can have varying causes. These irregularities can affect timetables, rolling stock and crew planning. For example, planned maintenance can change timetables and train routing. Train routing can require crew schedule changes because of the route knowledge requirements of train drivers. Short-term train orders can add train rides to the schedule, also requiring changes to the rolling stock and crew schedules.

Every Thursday, STP releases the Week Plan (WP). The release of the WP, does not mean the planning is now fixed. The plan changes up until the execution of the plan. Every day there is the Day Plan (DP), the plan for the upcoming day, which contains the latest changes. During the day of execution, all schedule changes are handled by the Real Time Management (RTM) department.

The company uses several Key Performance Indices (KPIs) to determine schedule efficiency. These are visualised per planning phase in figure 1.3. The KPIs are based on the percentage of time drivers are assigned different tasks in their shifts, excluding breaks. The KPIs correspond to the action categories as given in section 1.3.1. The KPIs are:

- **Productivity** Sum of time spent on *Train* and *Productive Actions*.
- **Train** Time spent driving trains.
- **Productive Actions** Time spent on all actions required to drive trains, such as inspecting, starting and shunting.
- **Move** Time spent relocating drivers without driving a train.
- **Idle** Time spent waiting for the next planned activity.

Figure 1.2 shows that from Week Plan to Day Plan, taxi costs increase by ∼45%. This is the steepest increase between any of the planning phases. This suggests that many schedule changes happen in the period between WP and DP. Figure 1.3 shows that the time spent on productive actions and driving trains barely changes in this period. The most significant difference is an increase in driver movements and a decrease in idle time. This means that for a similar amount of train time, more driver movements are used. Trains could be rescheduled, which causes conflicts in the driver schedules. Rescheduling drivers for the adjusted train timetables can create these additional movements. With this increase in driver movements, comes an even stronger increase in taxi costs.

From the RP to the WP, the time spent riding trains decreases. This is due to the cancellation of trains. The strong decrease in train time causes the driver movement time to decrease. Fewer trains that are driven means that fewer driver movements are required to reach the start location of train rides. This increases idle time, as the total working time does not change.

1.3.4 Real-time management

There are several departments involved in real-time management. During their shifts, train drivers are often in contact with these departments to confirm their current activities. Confirming that drivers have left and reached their locations according to schedule is part of guiding the operational process. The Real Time Management (RTM) department, that manages disruptions by rescheduling drivers and trains, is the RTM department related to this research.

RTM is tasked with solving all operational rescheduling problems. There can be many possible reasons why the original schedule becomes infeasible during operations. Train drivers could be late, either due to arriving late at work or because the previous train they were driving was late. It could also be that a company car is

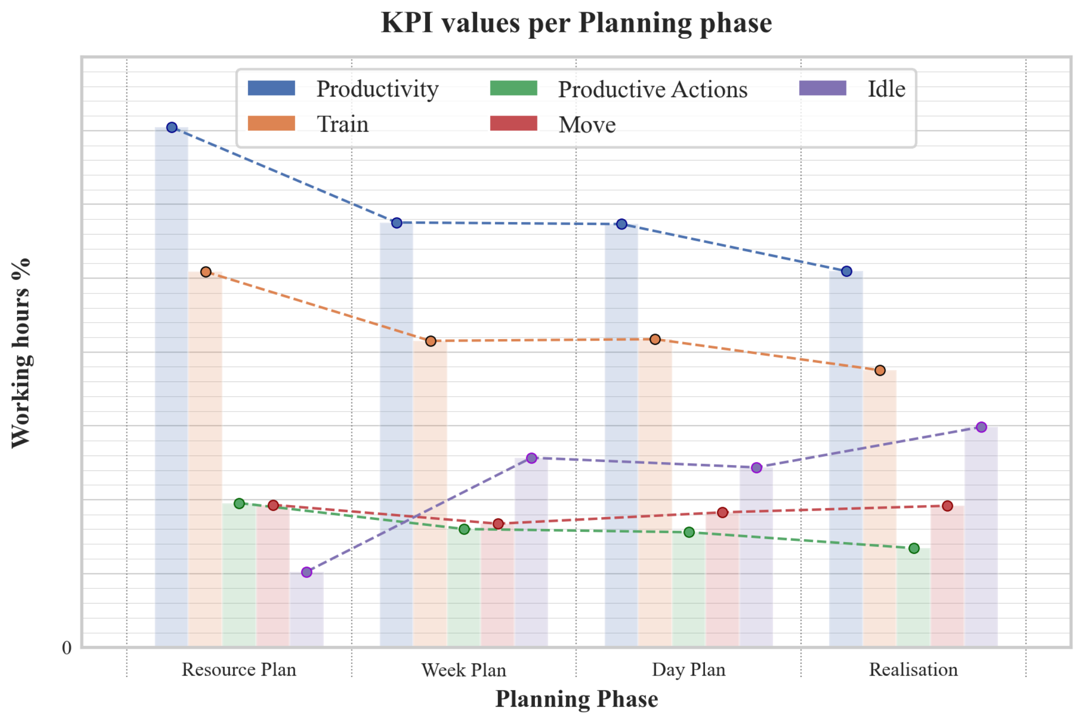

FIGURE 1.3: KPI values per planning phase

unavailable at a location because it has been moved by real-time management in the previous days. DBc's customers can also be causes of delays. They can be late delivering carriages. The unavailability of infrastructure is also a large problem. Tracks can be unexpectedly inaccessible for many reasons. This unavailability can be of short duration, causing delays, or can be of longer duration, possibly requiring re-routing of trains. All of these causes and any other cause not described here require the RTM department to create a solution.

Figure 1.3 shows that between the DP and realisation, the time spent driving trains and performing productive actions decreases. This suggests that some trains are cancelled during the day of operations. The decrease in productivity causes the idle time to increase. Even though fewer trains are driven, the driver movement time increases in the period between Day Plan and realisation. The taxi costs show a stronger increase (figure 1.2).

1.3.5 Real-time rescheduling process

The rail infrastructure is owned by ProRail and usage is shared by many train operating companies. When any problem happens, it is always up to the infrastructure owner (ProRail) to provide an updated timetable. ProRail aims to balance the interests of all railway-using companies when rescheduling the timetables. It is up to the rail operating companies to make the updated timetable feasible with their rolling stock and crew schedules. To accomplish this, they can: exchange rolling stock; shift crew schedules by exchanging tasks; and move train drivers to new locations. When exchanging tasks between train drivers, they have to consider driver-specific route knowledge and shift rules such as overtime and breaks. They also make sure that drivers end their shifts at their crew base. Figure 1.4 gives an overview of the real-time rescheduling process.

When DBc itself is unable to adhere to its appointed timeslots, they have to cancel and request new timeslots from the infrastructure owner. When requesting new timeslots, the infrastructure owner has no obligation to provide or accept the requested timeslots. The main responsibility of RTM is to make the rolling stock and crew schedules work with the timetables as proposed by the infrastructure owner. When DBc's dispatchers have found feasible new crew and rolling stock schedules for the new timetables, train drivers are updated with their new schedules.

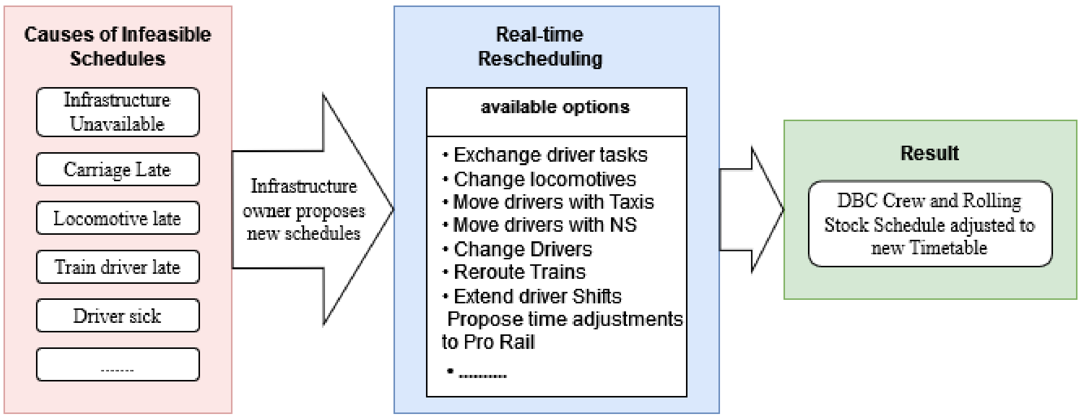

FIGURE 1.4: Current real-time rescheduling process

While DBc has no software that automatically calculates schedules, there are information systems and data available to help their decision-making. Through these systems, RTM has access to the current situation (disruptions) of the infrastructure. Operational information about the trains can be acquired through the Process Control Department (PROCO). All driver and rolling stock schedules can be retrieved from DBcs IT system, PALOMA. Timetables of the whole rail network can be retrieved through the systems TROTS and VKL. Another, significant method of data collection is to manually call people. Using the information available, the schedules have to be adjusted so trains reach their destination and drivers end their shifts at their crew base without too much overtime. Most times, a solution is required in minutes because many problems occur during normal operations.

1.3.6 Taxi costs

Because taxi costs have exceeded budget for several years, F&C has analysed the taxi costs. Figure 1.2 results from their analysis and shows that the Day Plan has the largest taxi cost increase. Their analysis also showed that through active management of taxi rides and company cars at some of their lines, taxi costs are lowered by 66% on those lines. Another finding was that there are routes which contain a high taxi costs. F&C believes that taxi costs can be reduced by adding company cars for these routes specifically. Most taxi costs are incurred in schedules from the *international* crew bases. These crew bases contain the drivers that ride international trains. International trips are often longer than national trains. Taxi rides from a crew base to the start or end location of an international train ride are typically expensive.

1.4 Problem Identification

This section defines the core problem that has to be solved. A problem cluster is created based on the analysis of the current situation in section 1.3. It shows what core problems contribute to the experienced taxi costs.

1.4.1 Problem cluster

Figure 1.5 shows the problem cluster. Disruptions are the biggest cause of the taxi cost problem. Disruptions represent all the reasons why the original schedule has to change. Without disruptions, the RP can be followed exactly, and operational costs would not increase. However, disruptions are inherent to railway operations and cannot be fully avoided. It is thus not chosen as the core problem, as it is not in our control. Because the RP is made manually, it could be optimized towards resilience to reduce the effect of disturbances. While it should not be disregarded that half of the realised taxi costs already exist in the RP, this department is not part of this research. Therefore, the lack of optimisation tools for the RP is not chosen as the core problem.

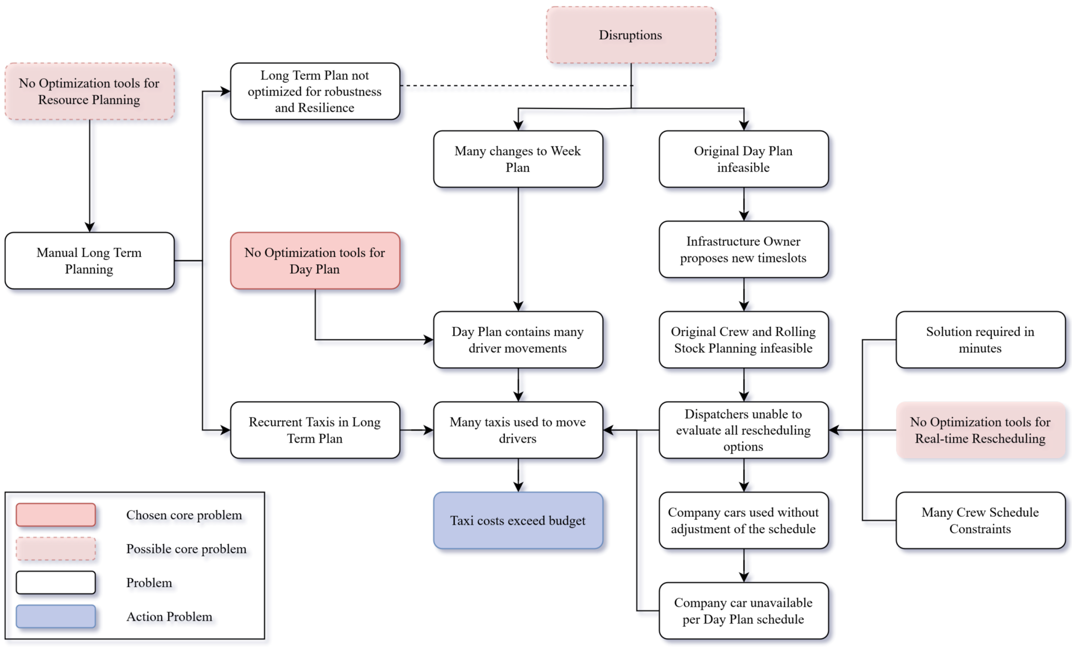
Disruptions
No Optimization tools for Resource Planning
Long Term Plan not optimized for robustness and Resilience
Many changes to Week Plan
Original Day Plan infeasible
Manual Long Term Planning
No Optimization tools for Day Plan
Infrastructure Owner proposes new timeslots
Day Plan contains many driver movements
Original Crew and Rolling Stock Planning infeasible
Solution required in minutes
Recurrent Taxis in Long Term Plan
Many taxis used to move drivers
Dispatchers unable to evaluate all rescheduling options
No Optimization tools for Real-time Rescheduling
Taxi costs exceed budget
Company cars used without adjustment of the schedule
Many Crew Schedule Constraints
Company car unavailable per Day Plan schedule
Chosen core problem
Possible core problem
Problem
Action Problem

FIGURE 1.5: Problem Cluster

The many changes that have to be made to the schedule during the last week before execution cause the Day Plan to contain many driver movements, resulting in many taxi rides. The KPIs per planning phase also show that productivity barely changes from Week Plan to Day Plan, while driver movement increases. This suggests that the schedule's efficiency has decreased, as more driver movements are being used for the same amount of train time. The schedule changes make the resulting DP much less efficient than the WP. There are currently no optimisation tools to increase the DP's efficiency.

During operations, disruptions require the Real Time Management department to solve infeasible crew and rolling stock schedules. These problems have to be solved quickly while considering all restrictions such as driver route knowledge, overtime, and breaks. The lack of decision-support systems makes it difficult to evaluate all options. This leads to the increase in taxi costs from Day Plan to realisation. RTM sometimes schedules company cars to prevent taxi costs, but due to the time requirement, they are unable to correctly implement the change in the company car's schedule. This can cause company cars to be unavailable later when they should be available according to the schedule. This results in an unplanned taxi ride, further increasing taxi costs.

1.4.2 Core problem

The causes of the taxi costs exceeding budget are coloured *red* in figure 1.5. Both the lack of decision support tools for real-time rescheduling and the inefficiency of the Day Plan regarding driver movements are possible core problems. It is important that a proposed solution can be validated and compared to data from practice. DB has a copy of the driver schedules from each planning phase, and of the realisation. During real-time management, the causes of infeasible schedule and the solutions found are not recorded. As chapter 1.3.5 describes, the real-time rescheduling process is complicated with many restrictions and options that the RTM department has to consider. For a decision support or rescheduling tool to be worthwhile, it should include all possibilities and restrictions. The evaluation of such a tool in relation to the present rescheduling performance is a challenging task due to the absence of details regarding the occurrences of disturbances and their resolution.

The core problem: There is no optimisation tool for the Day Plan is more measurable. The KPIs such as productivity and driver movement can be compared between original Day Plan and optimized Day Plan. The lack of optimisation tools for the Day Plan is therefore chosen as the core problem to be solved. The driver movement KPI can be used as a measurable unit on which schedules can be optimised. The core problem is defined as:

There is no optimisation tool to optimise the Day Plan.

1.5 Research Objective

Driver movements and taxi costs are closely related. Drivers move around in passenger trains, company cars, and taxi rides. Taxi costs are determined by both the planned taxi rides and the company car schedules, which makes it difficult to optimize taxi costs. When a company car is not available, a taxi will be used instead. Moreover, a planned taxi ride instead of a company car might reduce total taxi costs by not relocating a company car to a less beneficial location. By reducing the total time spent on driver movements and smart scheduling of company cars, costly taxi rides can be avoided. The objective of this research is defined as:

To develop an optimisation model that reschedules the Day Plan to reduce driver movements and optimise the use of company cars.

An optimisation model should reschedule the Day Plan to reduce the driver movements required to perform all planned productive activities in the Day Plan. The Day Plan currently has the largest increase in driver movements and taxi costs of any planning phase. There is thus much potential for driver movement reduction in this phase. A proposed solution can be compared and validated with original Day Plans and the Day Plan's driver movement KPI. Implementing automatic rescheduling or optimisation tools can be costly, but this research provides insight into possible savings that can be achieved with such optimisation tools. Based on optimised schedules, company car scheduling strategies can be compared to optimise company car usage. Maximising company car usage helps reduce taxi usage.

1.6 Problem Approach

To solve the core problem, a 5-step process is shown in figure 1.6. The research questions in section 1.6.1 are grouped and ordered by these process steps.

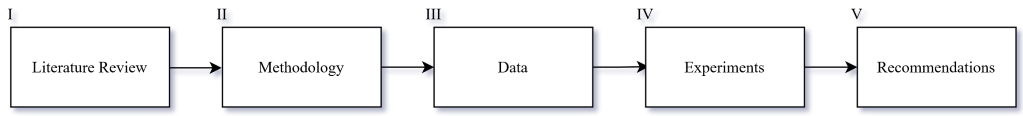

FIGURE 1.6: Problem Approach split into 5 steps

1.6.1 Research questions

This section discusses the structure of this thesis with accompanying sub-research questions. Following the objective and core problem of this research, the following question needs to be answered: *How can driver movements be reduced by designing and implementing a rescheduling tool that optimises the Day Plan and how do company car scheduling strategies influence company car utilisation?* To answer this question, the following sub-questions are formulated:

I **Literature Review**
Chapter 2 contains a literature review that provides insights into possible schedule optimisation models that best fit the research problem.

1. What new rescheduling model is best suitable for optimising the Day Plan?

II **Methodology**
Chapter 3 determines what should be in- and excluded in the optimisation model and how this will affect the results. It gives the mathematical representation of the rescheduling problem and develops the optimisation method.

2. What are the input, decisions and output of the rescheduling model?
3. What are the rescheduling model's limitations and how does this affect the practical applicability?
4. How must the mathematical model be adjusted to suit the problem?
5. What solving method should be applied to solve the model?

III **Data**
Chapter 4 discusses how the available data is used to generate Day Plans for the experiments. It also establishes a performance baseline using the generated datasets.

6. What data should be used to test the model?
7. What are the current Day Plan's KPIs?

IV **Experiments**
Chapter 5 discusses the experiments performed to evaluate the solution towards the research objective. It also provides the results of the performed experiments.

8. How much can driver movements be reduced?
9. How do different company car scheduling strategies affect company car usage?
10. How is taxi usage affected by reduced driver movements and company car scheduling strategies?

V **Recommendations**
Chapter 6 concludes the thesis with recommendations

11. What is required for the optimisation model to be implemented?
12. What can be recommended regarding company car scheduling?

2 Literature Review

Railway (re-)scheduling is a complex and widely researched topic. To create an understanding of the current state of research and possible solution methods for the research problem, this chapter provides an overview of related research, specified to (real-time) rescheduling. The method used for the structured literature review is explained in appendix B.1. At the end of this chapter, Research Question 1 (*What new rescheduling model is best suitable for optimising the Day Plan?*), is answered.

Most railway companies' (re-)scheduling processes consist of sequentially planned timetables (TT), rolling stock (RS) and crew (C) planning (Cacchiani et al. 2014). These subjects are used to categorize the related works in table 2.1. As section 1.3.5 describes, timetable rescheduling is the infrastructure owner's responsibility. For the rescheduling problem in this research, crew, and rolling stock rescheduling are the most important topics. Because DBc is allowed to propose timetable changes, timetable rescheduling is still relevant and therefore not excluded from the literature review. The literature regarding timetable rescheduling can be found in appendix B.2.

Section 2.1 discusses literature related to crew rescheduling. Afterwards, section 2.2 discusses literature related to rolling stock rescheduling. Section 2.3 discusses integrated rescheduling models combining one or more scheduling topics. Finally, section 2.4 gives a summary of the literature review and section 2.5 concludes this chapter by answering the research question 1.

2.1 Crew Rescheduling

Crew rescheduling is often the last part of rescheduling, but very relevant to the rescheduling problem in this research. The updated timetable and rolling stock circulation are typically given as input to crew rescheduling models. The goal is to create feasible crew schedules that cover as many timetabled activities as possible while adhering to crew scheduling laws for working hours and breaks (Cacchiani et al. 2014).

Rezanova and Ryan (2010) propose a train driver recovery model. The aim of *recovery*, is to return to the original driver schedule as quickly as possible. The model is formulated as a set partitioning problem, which is solved using a constraint branching strategy to find an integer solution to the Linear Program-relaxation. An algorithm specifically focused on rescheduling freight train drivers was developed by Sato and Fukumura (2010). From the adjusted timetable, they enumerate possible candidate duties. With the enumerated possible driver duties, the driver rescheduling is formulated as a set covering problem. They solved the model using a column generation algorithm. Column generation is a method often seen in crew scheduling. Potthoff et al. (2010) propose an algorithm using Lagrangian relaxation to solve the Operational Crew Rescheduling Problem (OCRSP). Like many crew rescheduling problems, they formulate it as a set covering problem. Column generation is typically used with larger disruptions, such as track blockages, on which many crew rescheduling problems focus. Existing crew rescheduling models perform much worse when handling smaller disruptions. Therefore, Verhaegh et al. (2017) developed a heuristic method to specifically focus on small disruptions. The heuristic outperforms the column generation approach when provided with a scenario of only one small disruption.

Besides column generation and set covering problems, other methods have been successfully used for crew rescheduling. Maekawa et al. (2020) use a zero-suppressed binary decision diagram to quickly enumerate different feasible crew schedules. This allows decision-makers to find suitable crew schedules that incorporate preferred conditions. The advantage is that the dispatchers can choose the most suitable alternative based on the situation, which allows them to utilize their expertise and experience. However, when networks become too large, the method struggles to find feasible solutions. Abbink et al. (2010) developed a unique method using an Actor Agent system in which independent actors form teams in which tasks are exchanged. This allows the model to use all available crew members instead of a subsection. Their Actor-agent model can generate new feasible schedules.

Vaidyanathan et al. (2007) perform crew scheduling on a Multi-Commodity Network Flow (MCNF), where different crew types are considered different commodities. Wang et al. (2022) propose a network flow model crew scheduling method based on a time-space-state network (TSSN). For duty generations, they use a classic shortest path problem. Meal breaks added in duty generation increase the complexity of the resulting set covering problem. Their Lagrangian relaxation approach generates good solutions quicker than traditional column generation approaches.

2.2 Rolling Stock rescheduling

Disruptions often cause the infeasibility of the original rolling stock allocation. Rolling stock units can be unable to carry out certain tasks in their duties. In the Rolling Stock Rescheduling Problem (RSRP), rolling stock is assigned to a set of trips in the timetable. When rescheduling rolling stock, the updated timetable is typically assumed to be given, together with the original rolling stock schedule (Cacchiani et al. 2014).

Sato and Fukumura (2012) developed a model that focuses specifically on freight locomotive rescheduling. They reassign locomotives to trains and then solve an uncovered train detection problem. Both linear programs are solved with column generation. The rescheduling uses relaxation of the set-partitioning constraints to speed up the solving. Malucelli and Tresoldi (2019) use column generation to solve a directed acyclic graph with a node set A partitioned into a set of drivers, services and special nodes to manage personnel breaks. The set of arcs A represents the compatibility between pieces of service.

The problem of rolling stock rescheduling can be formulated as a multi-commodity flow model on a graph. The nodes in the graph correspond to stations at specific times, while the arcs correspond to trips that have to be performed according to the

time-
table (Cacchiani et al. 2014). Nielsen et al. (2012) solved the RSRP with a rolling horizon, which means that the rolling stock is only rescheduled for a certain horizon. It is logical to assume that a new issue will arise in the future, which will require rescheduling again anyway. This allows future disruptions to be ignored, which will reduce computation times. To solve the RSRP they use a multi-commodity flow model in a graph constructed from the time-space diagram of the timetable. The model decides which compositions are assigned to each trip. Lusby et al. (2017) defined a path-based mathematical formulation for the rolling stock rescheduling problem (RSRP). According to the authors, the path-based model is more easily extended with additional constraints than flow-based approaches. The large integer program is solved using the branch-and-price technique. To generate columns, they generate feasible sub-trips using time-space networks. Because the inclusion of more details often increases complexity, Hoogervorst et al. (2021) propose a Variable Neighbourhood Search (VNS) algorithm to solve the RSRP. They develop 3 custom neighbourhoods for their VNS. They compared their model to an exact solution and found that their VNS performs comparable to exact calculation but can outperform exact calculation on larger and more detailed instances.

The rolling stock re-balancing problem (Budai et al. 2010) is a specific goal of rolling stock rescheduling, which aims to solve off-balances in the train units that end at each station. They use a network flow model to represent the railway.

2.3 Integrated rescheduling models

The literature discussed so far focuses either on crew or rolling stock rescheduling. Integrating different parts of the scheduling process can be beneficial. Sato et al. (2009) developed a linear programming formulation combining rolling stock rescheduling with crew rescheduling. For this, they used a multi-commodity network flow model. The advantage of using the network flow model, compared to set partitioning/covering models, is the ability to represent the differences between the new and original schedules. Sato et al. (2010) have also solved this problem using Lagrangian Relaxation. This comes with the added advantage that the quality of the solution can be determined based on the lower bound. Zeng et al. (2018) showed that a MCNF can also be solved to near optimality using a customised ant colony optimization algorithm.

Walker et al. (2005) created a model to simultaneously reschedule trains and crews. The railway system used is one in which single-track sections are common. They thus include a train crossing problem in their train rescheduling model. They solve it with a branch-and-bound algorithm using column and constraint generation. The model could theoretically create integrated schedules from scratch but is used to only generate schedules for trains and crews affected by disruptions.

Most crew rescheduling models reschedule up to the end of the day. The integrated crew re-planning problem (ICRPP) combines crew scheduling and crew rostering. Crew scheduling assigns tasks to crew shifts, while crew planning assigns those shifts to crews over multiple days. Crew planning considers working laws such as minimum resting time between shifts. Breugem et al. (2022) propose a column generation approach for the ICRPP. The combination of scheduling and planning means that the tasks are rescheduled for multiple days instead of the traditional up to the end of the day horizon. In recent work, Breugem et al. (2022) use column generation to solve an ICRPP. They use flexible shift end times and rostering constraints to enforce minimum rest times. Maenhout and Vanhoucke (2018) also use a longer planning horizon. They combine crew shift and task rescheduling with the uncertainty of capacity, demand, and arrival. Stochastic schedule disruptions are included using Monte Carlo simulation. The goal is to minimize the deviation in crew shifts compared to the original shifts. A perturbation meta-heuristic has been developed which outperforms a variable neighbourhood search and the CPLEX solver on larger instances. Ernst et al. (2001b) propose a crew scheduling method that also schedules multiple days and aims to reduce the size of the required workforce.

2.4 Summary

This chapter gave a broad overview of rescheduling models for train rescheduling. Rescheduling models can be categorised into timetable (TT), rolling stock (RS) and crew (C) rescheduling models. Some models try to incorporate these planning phases. Table 2.1 gives an overview of the literature discussed in this chapter, categorised on these topics.

The schedules of trains and locomotives are not directly related to DBc's taxi costs. Company cars, however, can be considered a type of rolling stock. Like locomotives, company cars are a resource required for an action. Locomotives are needed to move a train, and company cars are required to move a train driver. Both locomotives and company cars need a train driver to move. At DBc, company cars have a schedule, just like locomotives. Rolling stock rescheduling can thus be used for company car scheduling.

There are several models to represent railway crew rescheduling such as set covering, multi-commodity flow networks and time-space-state networks. Many crew rescheduling models are formulated as a set covering problem (Cacchiani et al. 2014; Ernst et al. 2001a). Set covering crew rescheduling models are often solved using branch-and-price (Borgonjon and Maenhout, 2022), Lagrangian relaxation (Veelenturf et al. 2012) or column generation (Huisman, 2007; Potthoff et al. 2010; Breugem et al. 2022). Disadvantages of the column generation method are that it is difficult to find the optimal solution within a practical time, and it is time-consuming to obtain solutions for actual problems (Muroi et al. 2010).

The integration of crew rostering and crew scheduling (ICRPP) (Ernst et al. 2001b; Maenhout and Vanhoucke, 2018; Breugem et al. 2022) is not relevant to the optimisation of the Day Plan. The ICRPP includes rest time constraints between shifts, which allows adjustment of the start and end times of shifts. This research considers rescheduling the Day Plan. Because it is not possible to extend a driver's shift one day in advance, both start and end times are fixed. The ICRPP is more useful when rescheduling for a longer planning horizon.

Article	Problem	Model	Railway representation	Subject	Solving method
Potthoff et al. 2010	C	Operational crew rescheduling problem (OCRSP)	Set Covering	Tasks	Heuristic with Column Generation and Lagrangian heuristics
Sato and Fukumura, 2010	C	Train Driver Rescheduling	Set Covering	Network of legs, drivers, and candidate duties	Column Generation
Breugem et al. 2022	C	Integrated Crew Replanning Problem (ICRPP)	Integer Linear Programm	Tasks, Duties	Column Generation, depth-first tree search
Abbink et al. 2009	C	Train Driver Rescheduling	Actor Agents	Tasks	Actor Agent
Maekawa et al. 2020	C	Flexible Crew Rescheduling Problem	Zero Suppressed binary decision diagram	Tasks	Gurobi
Abbink et al. 2010	C	Train Driver Rescheduling	Actor Agents	Tasks	Actor Agent
Verhaegh et al. 2017	C	Crew Rescheduling Problem (CRP)	Depth-first tree search	Tasks	Heuristic
Yuan et al. 2022	C	Flexible Crew Rescheduling Problem	Depth-first search crew recovery	Tasks	Algorithm
Ernst et al. 2001a	C	Crew rostering	Several Algorithms	Trips, stations, crew bases	Several
Huisman, 2007	C	Crew Rescheduling Problem (CRSP)	Set Covering	Tasks	Column Generation
Muroi et al. 2010	C	Crew Scheduling Problem (CSP)	Set Covering	Task	Column Generation
Vaidyanathan et al. 2007	C	Crew Scheduling Problem (CSP)	Multi Commodity Flow Network	Tasks	Relaxation, Decomposition
Wang et al. 2022	C	Crew Scheduling Problem (CSP)	Time space state network	Trips, stations, crew bases	Lagrangian Relaxation
Ernst et al. 2001b	C	Crew Scheduling and Rostering	Sequential Scheduling and Rostering	Trips and Rosters	CPLEX
Hoogervorst et al. 2021	RS	Rolling Stock Rescheduling Problem (RSRP)	Directed acyclic graph	Trips, transitions	Variable Neighbourhood Search
Lusby et al. 2017	RS	Rolling Stock Rescheduling Problem (RSRP)	mixed integer linear programming	Stations, depots, unit types, arrival events	Branch and Price
Nielsen et al. 2012	RS	Rolling Stock Rescheduling Problem (RSRP)	Constraint multi-commodity flow model	Trips, rolling stock types, compositions	CPLEX
Sato and Fukumura, 2012	RS	Locomotive Rescheduling	Set Partitioning	Tasks	Column Generation
Budai et al. 2010	RS	Rolling Stock Rebalancing Problem (RSRP)	Network Flow Model	Trips and Stations	2 phase heuristic
Sato et al. 2009	RS + C	Crew rescheduling problem + Vehicle rescheduling problem (CRP/VRP)	Network Flow Model	trips, transition	construct + local search
Zeng et al. 2018	RS + C	Integrated rolling stock and crew rescheduling (IRSCRP)	Integer Linear Program	Multi-commodity flow model	Ant Colony
Malucelli and Tresoldi, 2019	RS + C	Crew Rescheduling Problem (CRP)	Set Partitioning	Directed acyclic graph, Compatibility graph (tasks and train duties)	column generation
Sato et al. 2010	RS + C	Crew rescheduling problem + Vehicle rescheduling problem (CRP/VRP)	Network Flow model	Trains, stations, events	Lagrangian Relaxation

TABLE 2.1: Overview of real-time railway rescheduling models.

Sato et al. (2010) model crew and rolling stock rescheduling on a Multi-Commodity Network Flow (MCNF). The MCNF has the advantage of being able to reschedule both crew and rolling stock. This is useful for the rescheduling problem in this research, as it allows both schedules of crews and company cars to be represented on a single network model. Sato et al. (2010) solved the model using Lagrangian relaxation as well as with a local search, (Sato et al. 2009). Because they aim to solve the crew and rolling stock schedules for disruptions in real-time management, the starting point is a flow conflict. Their local search method could be applied to optimise the Day Plan by not starting with flow conflicts but with the longest duration taxi rides.

2.5 Conclusion

Based on the literature review, this section concludes the chapter by answering the research question 1: *What new rescheduling model is best suitable for optimising the Day Plan?*

A Multi-Commodity Network Flow will be used for rescheduling the Day Plan as this allows both the crew and company cars to be represented on the same network. Hoogervorst et al. (2021) Successfully applied a Variable Neighbourhood Search (VNS) to solve the Rolling Stock Rescheduling Problem represented on a MCNF. Because the rescheduling problem in this research is similar to that of Hoogervorst et al. (2021) and is represented on a similar network, a VNS is likely to find adequate solutions. Therefore, a VNS will be used to optimise the Day Plan. A VNS also extends the previous local search method by Sato et al. (2009), which successfully generated feasible schedules.

The gap in the literature that is addressed is the lack of crew rescheduling models focussed on reducing driver movements. Crew rescheduling literature generally focuses on crew scheduling costs such as overtime or reducing the required workforce (Ernst et al. 2001b). A focus on driver movements is especially interesting for the rail freight sector because train trips are longer, less frequent, and less cyclical than passenger trains. As a result, driver movements are more frequent, and when taxis are used to move drivers, these movements can be quite costly. The next chapter (3) discusses the methodology.

3 Methodology

The methodology to accomplish the research objective is divided into three steps. The sections related to the three steps are shown in table 3.1. The first step (**1**) optimises the Day Plan and consists of the solution design choices (section 3.2), the mathematical modelling (section 3.3) and the optimisation method (section 3.4). The second step (**2**) aims to optimise company car usage. Using optimised Day Plans, the effect of different company car scheduling strategies on company car utilisation is compared (section 3.5). The last step (**3**) focuses on the combined effect of reducing driver movements and company car scheduling on taxi usage (section 3.6). Before that, the very first section of this chapter (3.1) discusses what data is available in this research.

	Optimising the Day Plan			**Company car scheduling**			**Evaluating Taxi Costs**
1	3.2: Solution Design 3.3: Mathematical Modelling (MCNF) 3.4: Optimisation Method (VNS)	$\Rightarrow$	**2**	3.5: Comparing company car scheduling strategies	$\Rightarrow$	**3**	3.6: Combining Day Plan optimisation with company car scheduling to evaluate the effect on taxi costs

TABLE 3.1: 3-Step Methodology process.

3.1 Available Data

This section discusses how DBc stores their data and what data is available in this research. An example driver schedule is created using the provided data. This section also discusses how anonymity limits the available data. Finally, a scope is set for the data used in this research.

3.1.1 Driver shift data

Because of the company's long existence and natural growth, the company's data is stored in several (disjoint) systems. Train data is separated from driver data. DBc stores combined historical data in a large database. Only a part of the data in this database is available in this project. The available data corresponds to the data from DBcs planning software PALOMA, which contains every planned driver action. This driver data exists for each planning phase of 2022. Planned actions contain an identifier of the order to which they belong. An order is a group of train trips belonging to a single train. During a shift, train drivers can perform actions on multiple trains. A driver shift includes all the individual tasks that the train driver is required to execute on that particular day.

The combination of shift and day is an identifier with which individual driver schedules can be recreated. Figure 3.1 shows an example schedule. The horizontal bars plotted over time represent the actions the driver is assigned to perform. Actions with a number stand for a train movement, while other actions are abbreviated. The definitions of the actions in the figure 3.1 are given in table 3.2. The actions planned in this driver shift are just an example of the many possible actions that can be assigned to drivers. Every driver schedule is assigned to a crew base. For every crew base, several starting locations can be determined from the shift identifier.

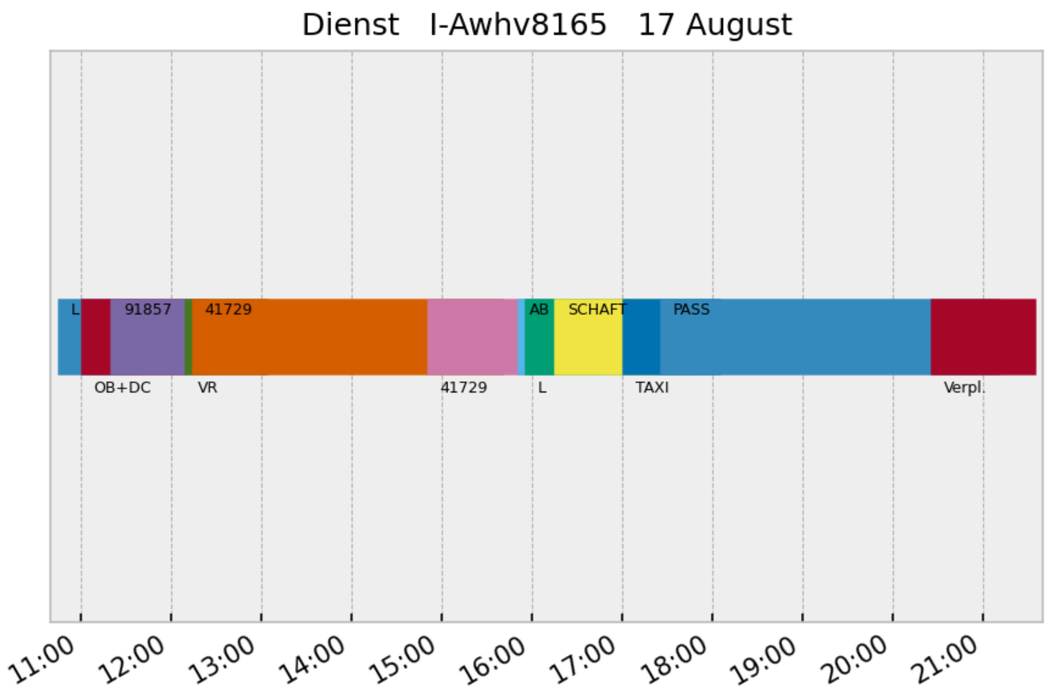

FIGURE 3.1: Example driver schedule on the 17th of August 2022

Action	Description	Category
L	Walking	Move
OB+DC	Preparing locomotive + Daily check of locomotive	Productive
91857, 41729	Order number representing train ride	Train
VR	Shunting	Productive
AB	Power down locomotive	Productive
SCHAFT	Break	Break
TAXI	Taxi ride	Move
PASS	Passengering (possibly other DBc train, or public transport)	Move
Verpl.	Driver move (any method)	Move

TABLE 3.2: Description of planned actions in figure 3.1.

3.1.2 Data limitations

The available data is anonymous and does not show which driver a shift is assigned to. At DBc, drivers have individual route knowledge, which specifies the routes they are certified to ride. Similarly, the material knowledge of drivers determines which locomotives they are certified to operate. The driver-specific route and material knowledge cannot be used in this research because of the anonymity of the available driver data. Because of the anonymity, it is also impossible to determine which crew performed which schedules on consecutive days.

3.1.3 Data scope

The whole planning of DBc is very complex. There are many activities and locations. Furthermore, it contains 3 categories of employees: *local*, *national* and *international*. As per section 1.3.6, international shifts contain the most taxi costs. Local shifts are managed separately, and national shifts contain more activities requiring less driver movements, such as shunting or shorter train trips. Because international shifts contain the most taxi costs, it is decided to focus this research on international shifts only. The locations and driver actions are also selected based on locations that occur in international shifts. Productive actions that are not train trips, and happen at a single location, can be simplified without loss of generality. The names of the actions are not relevant to the optimisation of the schedule, only the durations. Chapter 4 goes into more detail about the data used for the experiments.

3.2 Solution Design

This section discusses the specific design choices of the solution. With that, it answers research questions 2: *What are the input, decisions and output of the rescheduling model?* and 3: *What are the rescheduling model's limitations and how does this affect the practical applicability?*. Section 3.2.1 describes the input, decisions, and output of the model (RQ2). The limitations of the model and their influence on the practical applicability of the model (RQ3) are discussed in section 3.2.2.

3.2.1 Model requirements

This section discusses what is included in the rescheduling model. The required input, the possible decisions, the expected output, and the assumptions are described.

Input The model is given the Day Plan existing of all driver schedules of international crew bases. Each driver's schedule has a start location with a start time and an end location with an end time. All train trips and related activities, such as shunting assigned in the Day Plan, make up the *productive* actions. These are all actions that have to be included in driver schedules. Every driver's schedule contains a 30-minute break. A document containing the duration of taxi trips between locations is also available. This can be used for planning new taxi rides.

Decisions The model should reschedule the crew to reduce driver movement time. All productive actions have to be assigned to driver shifts. The model decides which productive actions are assigned to which driver shift. The model can change, remove or add non-productive actions such as driver movements to make the shifts feasible.

Output The model should return a feasible schedule in which all productive actions have been added to a driver shift. A feasible crew schedule has the following constraints: The start and end locations and times must be the same as the original schedule; it should contain a 30-minute break, and the flow of actions should be correct. The flow of a driver's schedule is correct if the start location of every action is the same as the end location of the previously planned action.

Assumptions The start and end times of productive actions can't be changed, as this would change the timetable, which requires intervention of the infrastructure owner. Changes to the start and end times of driver shifts are not permitted. The original shift's starting point can't be altered, and the shift must end at the original location. This is important as it forces the drivers to return to the starting location at the end of their shift. The start time of breaks can be changed, but the duration can not. The schedule of locomotives is disregarded, as locomotive schedules are not available. Planned train rides are assumed to have a locomotive assigned.

3.2.2 Model limitations

Because railway scheduling is very complex, a rescheduling model requires a selection of details that are in- and excluded. This section discusses the limitations of the model and how they affect its practical applicability.

Passenger trains are a suitable and cheap method to move drivers and are used a lot by DBc. To schedule passenger train trips, the optimisation tool would require all locations of passenger train stations and all passenger train timetables. Furthermore, it would require route calculation from every location to the nearest passenger train station. This can be achieved by implementing the NS API into the optimisation model. The API has both route calculation from and to train stations, as well as route calculation of public transport. Because of the resulting complexity increase, it is decided that implementing the NS API is beyond the scope of this research. Therefore, driver movements have to be simplified. 3 driver movement terms are used: *TAXI*, *TAXI/DA* and *DA*. DA is a company car and TAXI/DA means a driver should take a company car when one is available and a taxi otherwise. TAXI allows the model to specifically assign taxi rides even when a company car is available at a location.

The removal of passenger train trips has a big effect on the results of the optimisation model. Taxi usage will be much higher without passenger train trips. The rescheduling model's performance thus cannot be determined based on taxi costs. Instead, the driver movement time KPI can be compared because both passenger train trips, and taxi rides, are driver movements. Would DBc implement the optimisation tool, then passenger train trips must be added as a driver movement option by implementing the NS API. The MCNF uses *nodes* to represent planned actions, such as train trips and driver movements. This is further explained in section 3.3. Route calculation from any location to the closest train station is included in the NS API, as well as from train stations to other stations. Each driver's movement using passenger transport can thus be represented by a node in the MCNF. Using the API, it would be possible to generate new nodes (trips) when needed instead of calculating every possible trip using public transport in advance.

In Real Time Management, shifts can be extended with overtime as a last option if it fits within labour laws. Because the data available in this research is anonymous, driver schedules on consecutive days cannot be linked to individual drivers. Labour laws such as minimum rest time thus cannot be guaranteed if the end time of a shift is changed. Therefore, the optimisation model cannot adjust the start and end times of shifts. This reduces the optimisation capabilities, but ensures compliance with rostering rules and labour laws.

Because route and material knowledge is train driver-specific and crew schedule data is anonymous, this knowledge cannot be used in the rescheduling model. While many drivers from international crew bases have similar route knowledge, this is not always the case. The removal of the route knowledge restriction results in a slight relaxation of the rescheduling model. Would DBc implement a rescheduling tool, then driver-specific route knowledge should be added. This is possible as the data exists, it is just not available in this project. Locomotive knowledge is also driver-specific, but most international routes use the same locomotives. Therefore, it can be disregarded without affecting the practical applicability too much.

3.3 Multi-Commodity Network Flow

The combined rolling stock and crew rescheduling problem proposed by Sato et al. (2010) based on a Multi-Commodity Network Flow (MCNF) is well suited for the rescheduling problem in this research. It does require some adjustments. This section starts with an explanation of the MCNF. Then the required changes are discussed, answering research question 4 *How must the mathematical model be adjusted to suit the problem?* The new mathematical model is presented in section 3.3.2.

A train timetable can be partitioned into *trips*. Trips are movements between locations where resources can be exchanged. A *resource* refers to a vehicle or crew. Each timetabled trip is represented in the network as a *node*. Nodes are linked with directed arcs if a resource can transition from one node to another node. A resource's schedule can be represented on the network as flows of the resource over the directed arcs. Figure 3.2 shows a representation of the MCNF.

A solid arrow represents a resource flow, which allocates a resource to both the nodes related to the arc. These are the node from which the arrow departs and the node where the arrow arrives. A series of solid arrows is a schedule, of a single resource. Resources can flow between nodes only if there is a directed arc between the nodes. For node A to have a directed arc to node B, the trip represented by node B must start after the trip represented by node A ends. The start location of B must be equal to the end location of A. A node can have resource requirements, which determine the number of resources that have to be assigned to flows including the node.

3.3.1 Mathematical formulation

Based on the MCNF described above, an integer programming formulation is given to the Crew Rescheduling Problem/Vehicle Rescheduling Problem in (1a) - (1h). The definition of sets, variables and parameters is given in table 3.3.

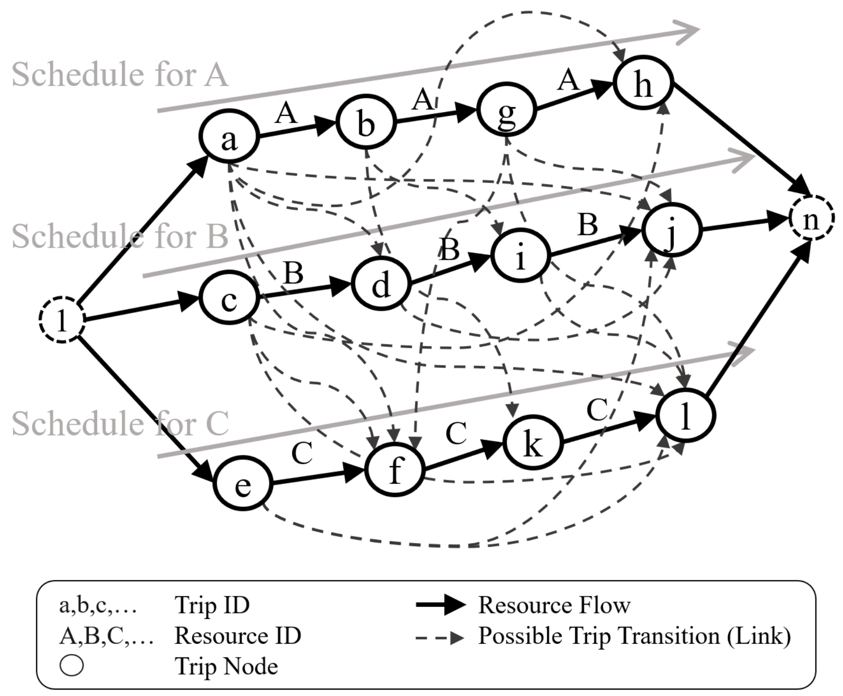

FIGURE 3.2: Multi-commodity Network Flow for Crew/Vehicle Schedules, *adjusted from:* Sato et al. (2009)

Sets	**Description**
V	Set of nodes $\{1, \cdots, n\}$ that consists of all the trips and dummy nodes
E	Set of directed links $\{a_{ij} \equiv (i,j) \mid i,j \in V\}$ such as (i 's end time) $<$ (j 's start time) and (i 's end location) $=$ (j 's start location)
R	Set of resources $\{1, \cdots, m\}$
Variables	
x_{ij}^k	Binary $(0,1)$ variable denoting the amount of resource r_k on the directed link $(i,j) \in E$
y_i	Integer variable denoting the amount of lack of resources on node $i \in V$
Parameters	
e_{ij}^k	Flow feasibility, that is, e_{ij}^k takes 1 if resource k is able to flow from i to j
$d_i (> 0)$	Number of resources required for node i
c_{ij}^k	Cost of resource k 's flow on link (i,j)
w	non-negative value that determines the weight of the variable y_i

TABLE 3.3: Description of sets, variables and parameters in equation (1a) - (1h).

$$\begin{aligned}
&\underline{\textbf{min}} && \sum_{k=1}^{m}\sum_{(i,j)\in E} c_{ij}^k x_{ij}^k + w\sum_{i=1}^{n} y_i && && \text{(1a)}\\
&\underline{\textbf{s.t.}} && \sum_{(1,j)\in E} x_{1j}^k = 1 && \forall k \in R && \text{(1b)}\\
& && \sum_{(i,n)\in E} x_{in}^k = 1 && \forall k \in R && \text{(1c)}\\
& && x_{ij}^k \le e_{ij}^k && \forall (i,j)\in E, \forall k \in R && \text{(1d)}\\
& && \sum_{(q,i)\in E} x_{qi}^k = \sum_{(i,q)\in E} x_{iq}^k && \forall i \in V\backslash\{1,n\}, \forall k \in R && \text{(1e)}\\
& && \sum_{k=1}^{m}\sum_{(q,i)\in E} x_{qi}^k + y_i = d_i && \forall i \in V\backslash\{1,n\} && \text{(1f)}\\
& && x_{ij}^k \in \{0,1\} && \forall (i,j)\in E, \forall k \in R && \text{(1g)}\\
& && y_i \in \{0,\cdots,d_i\} && \forall i \in N\backslash\{1,n\} && \text{(1h)}
\end{aligned}$$

The constraints are explained as follows:

(1a) The objective function minimises the cost based on flow costs c_{ij}^k multiplied by resource assignment to these flows x_{ij}^k and adds the cost of lacking resources y_i.

(1b) The first node 1, is a dummy node representing the start of any driver shift. This constraint ensures that each driver starts their shift in the dummy start node 1.

(1c) The last node n, is a dummy node representing the end of any driver shift. This constraint ensures that each driver ends their shift in the dummy start node n).

(1d) This constraint enforces that resources can only be assigned on a flow if that resource flow is feasible according to the directed arcs e_{ij}^k.

(1e) The flow conservation constraint enforces that resources should depart from a node if they arrive in that node (for all nodes, except the start 1 and end n node).

(1f) The variable y_i is set based on the resource assignment x_{ij}^k and resource requirement d_i and represent the lacking resources in each node.

(1g) Variable x_{ij} is a binary decision variable.

(1h) The values variable y_i can take on.

3.3.2 New mathematical formulation

This section answers research question 4: *How must the mathematical model be adjusted to suit the problem?* It describes the required changes to formulate the rescheduling problem in this research as a mathematical optimisation problem and explains the adjusted constraints.

The original model by Sato et al. (2010) uses a single start and end node. All schedules end and start in this node. For the rescheduling problem in this research, there are different starting locations. Every original driver schedule has a related start and end node containing the start and end time and location respectively. While drivers can start at the same time at the same location, each driver has their own dummy start and end note for notational convenience. Besides additional start and end nodes for driver resources, the model also needs some adjustments for company cars. Company cars have to be driven by a train driver which means that both a company car as well as driver resource must be assigned to a node if that node represents a company car trip. The company cars also have a dummy start node that represents their start location but the company cars can end their shift in any location. Therefore, a single dummy end node is used for all company cars (t_d in figure 3.3.

Figure 3.3 shows the adjustments of the MCNF in the graph representation. It shows the flow of company car resource D in red, as well as the new start and end nodes. The company car resource can only flow from and to nodes that are also assigned to a crew schedule. The new sets, variables and parameter definition are shown in table 3.4 and the new mathematical model is given in (2a) - (2k).

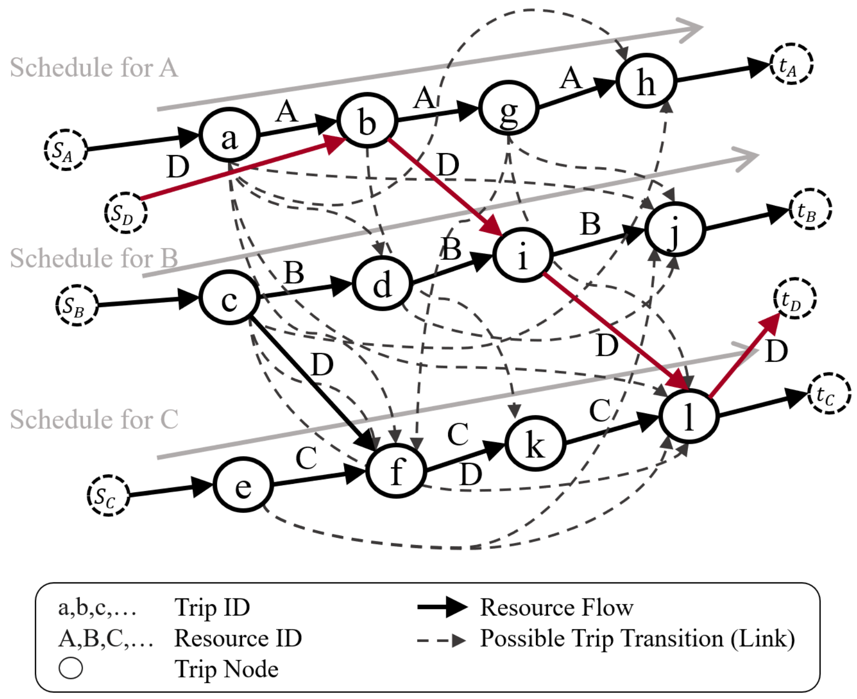

FIGURE 3.3: Updated Multi-commodity Network Flow for Crew/Vehicle Schedules

Sets	**Description**
V	Set of nodes $\{1, \cdots, n\}$ that consists of all the trips and dummy nodes. Trips include productive actions that have to be planned such as train rides, as well as all possible nonproductive actions such as taxi or company car trips.
v'	Subset of V with length k representing driver start nodes $v' = \{v_i \mid i \in \{1, 2, \ldots, k\}\}$
t'	Subset of V with length k representing driver end nodes $t' = \{t_i \mid i \in \{n-k+1, n-k+2, \ldots, n\}\}$
s'^p	Subset of V with length p representing start nodes of the company car resources.
t'^p	Subset of V with length 1 representing the arbitrary end node of company car resources.
E	Set of directed links $\{a_{ij} \equiv (i, j) \mid i, j \in V\}$ such as (i 's end time) $<$ (j 's start time) and (i 's end location) $=$ (j 's start location)
Rd	Set of driver resources $\{1, \cdots, k\}$
Rp	Set of company car resources $\{1, \cdots, p\}$
R	Set of resources $R = Rd \cup Rp$
Variables	
x_{ij}^k	Binary $(0, 1)$ variable denoting the amount of resource r_k on the directed link $(i, j) \in E$
y_i	Integer variable denoting the amount of lack of resources on node $i \in V$
Parameters	
e_{ij}^k	Flow feasibility, that is, e_{ij}^k takes 1 if resource k is able to flow from i to j
$d_i (> 0)$	Number of driver resources required for node $i \in V$
$p_i (> 0)$	Number of company car resources required for node $i \in V$
c_{ij}	Flow costs of link (i, j)
g_i	Trip duration of node (i)

TABLE 3.4: Description of sets, variables and parameters of the updated mathematical model.

$$\begin{aligned}
&\underline{\textbf{min}} && \sum_{k=1}^{k}\sum_{(i,j)\in E} c_{ij}x_{ij}^{k} + \sum_{i=1}^{n} g_i y_i && && \text{(2a)}\\
&\underline{\textbf{s.t.}} && \sum_{(s'_k,j)\in E} x_{s'_k j}^{k} = 1 && \forall k \in Rd && \text{(2b)}\\
& && \sum_{(i,t'_k)\in E} x_{it'_k}^{k} = 1 && \forall k \in Rd && \text{(2c)}\\
& && x_{ij}^{k} \le e_{ij}^{k} && \forall (i,j) \in E, \forall k \in Rd && \text{(2d)}\\
& && x_{ij}^{p} \le \sum_{k\in Rd} x_{ij}^{k} * e_{ij}^{k} && \forall (i,j) \in E, \forall p \in Rp && \text{(2e)}\\
& && \sum_{(q,i)\in E} x_{qi}^{k} = \sum_{(i,q)\in E} x_{iq}^{k} && \forall i \in V \setminus (v' \cup t'), \forall k \in Rd && \text{(2f)}\\
& && \sum_{(q,i)\in E} x_{qi}^{k} = \sum_{(i,q)\in E} x_{iq}^{k} && \forall i \in V \setminus (s'^{p} \cup t'^{p}), \forall k \in Rp && \text{(2g)}\\
& && \sum_{k=1}^{k}\sum_{(q,i)\in E} x_{qi}^{k} = d_i && \forall i \in V && \text{(2h)}\\
& && \sum_{k=1}^{p}\sum_{(q,i)\in E} x_{qi}^{k} + y_i = p_i && \forall i \in V && \text{(2i)}\\
& && x_{ij}^{k} \in \{0,1\} && \forall (i,j) \in E, \forall k \in R && \text{(2j)}\\
& && y_i \in \{0,\cdots,p\} && \forall i \in N\setminus\{1,n\} && \text{(2k)}
\end{aligned}$$

The new constraints allow the scheduling of both crew and company car resources. The original model allowed multiple resources, but only if they were all scheduled the same way. Because company car resources also require a driver resource to be scheduled, new constraints are needed for the scheduling of company car resources. The constraints are explained as follows:

(2a) The objective function calculates the cost based on flow costs c_{ij} multiplied by resource assignment x_{ij}^{k} and adds the cost of lacking resources y_i based on the nodes' trip duration g_i. The flow cost is usually 0, as the model does not aim to reduce crew use. However, if a flow contains a node that is a planned taxi ride the flow cost is non-zero.

(2b) Each driver starts their shift at their start node v'_k. The start node includes the start time of their shift and the start location. While drivers can start at the same location and time, each driver is given an individual start node.

(2c) Each driver ends their shift in their end node t'_k. It is used to ensure that all drivers end their shift in their start location without overtime.

(2d) Driver resources can only be assigned on a flow if that resource flow is feasible according to the directed arcs e_{ij}^{k}.

(2e) Company car resources can only be assigned on a flow if that resource flow is feasible according to the directed arcs e_{ij}^{k}. Because company car trips also require a driver resource as well, a driver resource must also be planned x_{ij}^{k}.

(2f) The flow conservation constraint enforces that driver resources should depart from a node if they arrive in that node (for all nodes, except the set of start v' and end (t') nodes).

(2g) The flow conservation constraint enforces that company car resources should depart from a node if they arrive in that node (for all nodes, except the set of start s'^p and end t'^p nodes).

(2h) Ensures assignment of driver resources x_{ij}^k to productive nodes according to the resource requirement d_i. Nonproductive actions such as driver movements do not have to be scheduled but can be used to adhere to the flow conservation constraint (2f).

(2i) Assigns the lack of company car resource to variable y_i according to the company car resource requirement p_i.

(2j) The resource assignment decision variable x_{ij}^k is a binary variable.

(2k) Sets the values that the lack of company car resource y_i can take on. There is no such constraint for lack of driver resources as all productive nodes must be scheduled in a driver shift. Lack of driver resources is not an option.

3.4 Optimisation Method

The complexity of the mathematical model (2a) — (2k) is dependent on the size (n) of the node set V, the resource set R and the set of directed arcs E. The size of the directed arc set E depends on the size of the node set V. In a complete graph, the number of possible arcs e in set E between the nodes $i, j \in V$ has the size of $n * n$. Because there is a decision variable x_{ij}^k for every arc e, the growth of the node set V leads to an exponential growth of the number of binary decision variables. The model can't just re-allocate existing nodes, it can also add new ones that weren't included in the original plan. The node set V thus contains all possible trips that can be planned. This includes every possible taxi ride. Taxis can leave from any place at any time, which means there can be an infinite number of taxi stop points in the set V. This causes exact calculation to be impossible without limiting possible taxi rides. A heuristic method is therefore required. This section answers research question 5: *What solving method should be applied to solve the model?*

The cost calculation in the mathematical formulation (2a) is dependent on both taxi rides and taxi rides due to lacking company car resources. The time of company car rides is not counted in the cost calculation. Because company car trips also require driver resources, the schedules of company cars and drivers are intertwined. If a company car trip is planned, but no company car is available, this trip becomes a taxi ride. This means that rescheduling drivers influences company cars and vice versa. This complicates movement time calculation in a heuristic. Reducing driver movements might increase the total calculated time because a company car isn't available. To solve this, the DP is first optimised regarding driver movements, but not considering company cars specifically. Then, based on the optimised schedules, company car scheduling strategies are compared. This allows the rescheduling model to evaluate crew schedules directly based on total movement time, without having to optimise company car schedules after each change in driver schedules.

Hoogervorst et al. (2021) Optimised a MCNF using a Variable Neighbourhood Search (VNS) with good results. Sato et al. (2009) applied a local search to solve their rescheduling problem. As a VNS is a metaheuristic based on local search operators, it is expected that a VNS can successfully be used to reduce driver movement time in the rescheduling problem as represented here on a MCNF. VNS is chosen as the optimisation method to reduce driver movements. Section 3.4.1 explains a general VNS. Section 3.4.2 shows the proposed VNS and section 3.4.3 describes the individual operators of the VNS.

3.4.1 Variable Neighbourhood Search

Algorithm 1 shows an example of a VNS. A VNS generally repeats the following steps: (i) select a random solution from a neighbourhood, (ii) perform a local search on the solution and finally (iii) change to a new neighbourhood. The randomness in the first step prevents cycling between solutions. The second step optimises the solution to a local optimum. The final step aims to escape local optima by changing the operator, which moves the search to a new solution neighbourhood (Hansen et al. 2010).

A VNS uses a *shake*, which is seen in line 5 of algorithm 1. This shake corresponds to the first step (i). The shake step prevents solution cycling and allows the VNS to escape local optima, by introducing randomness. This is often done by creating several solutions from a neighbourhood and selecting the best of these randomly generated solutions (Mladenović & Hansen, 1997a). Line 6 of algorithm 1 shows the local search step (ii) of the VNS. For the local search, a difference can be made between *first improvement* and *best improvement*. Using a first improvement method as in line 6 of algorithm 1 means the first found improvement is selected as the current solution. The best improvement method requires the calculation of the whole neighbourhood. The best solution in the neighbourhood would be selected. Finally, the neighbourhood changes (step iii) in line 7 of algorithm 1.

Algorithm 1: Variable Neighbourhood search (*adjusted from:* Hansen et al. (2010))

```
1 Function VNS($x, k_{max}, iter_{max}$);
2 while $iter \leq iter_{max}$ do
3     $k \leftarrow 1$
4     while $k < k_{max}$ do
5         $x' \leftarrow Shake$(x,
          _k) x″← $FirstImprovement$(x′) $NeighbourhoodChange$(x, x″, k)
```

Generally speaking, the neighbourhood used for local search is different from the other neighbourhoods (Mladenović & Hansen, 1997b). This means that the same local search operator is used on solutions from the different neighbourhoods. A variant of the VNS is the Variable Neighbourhood Descent (VND). The algorithm of a VND is shown in algorithm 2. Instead of a local search followed by a change of neighbourhood, the neighbourhoods themselves are used as the local search methods. Using the best improvement method, the best solution of the neighbourhood is selected as the current solution (line 5 of algorithm 2). Afterwards, the next neighbourhood is selected (line 6 of algorithm 2). This is repeated for every neighbourhood *k* (line 4), as long as improvements are found (line 2).

Algorithm 2: Variable Neighbourhood Descent (*adjusted from:* Hansen et al. (2010))

```
Function VND($x, k_{max}$);
while improvement found do
    $k \leftarrow 1$
    while $k < k_{max}$ do
        $x' \leftarrow argmin_{y \in N'_k(x)}$;
        NeighbourhoodChange($x, x', k$);
```

The combination of VNS and VND is also known as a General Variable Neighbourhood Search (GVNS). This approach has led to many successful applications (Hansen et al. 2010). The GVNS has the advantage of using different neighbourhoods as local search methods like the VND, while also using the VNS methods of shaking and changing neighbourhoods. A GVNS is also the type of VNS used by Hoogervorst et al. (2021) to optimise their MCNF. A general algorithm of the GVNS is given in algorithm 3. The GVNS uses a max running time (line 2). While this max running time has not been reached, it loops over the neighbourhoods k (line 4). A shake selects a random solution from neighbourhood k as the new solution (line 5). A VND (algorithm 2) is performed on the selected solution (line 6). This results in a local optimum based on the neighbourhood set k'. Finally, the neighbourhood k is changed (line 7).

Algorithm 3: General variable neighbourhood search (*adjusted from:* Hansen et al. (2010))

```
Function GVNS($x, k'_{max}, k_{max}, time_{max}$);
while $time \leq time_{max}$ do
    $k \leftarrow 1$
    while $k < k_{max}$ do
        $x' \leftarrow Shake$(x,k)
        $x'' \leftarrow VND$(x', k'$_{max}$);
        NeighbourhoodChange($x, x'', k$);
    $t \leftarrow CpuTime()$
```

3.4.2 Proposed variable neighbourhood search

The proposed solution is a GVNS. It allows the local search step of the algorithm to be replaced with a VND, while also adding the VNS processes of shaking and changing neighbourhoods. Because of the complexity of the planning, the algorithm needs to enforce schedule feasibility. Once the schedule becomes infeasible, it is unlikely to become feasible again. It is thus challenging to make a single local search method that can be applied to solutions from any neighbourhood. Replacing the local search with a VND allows for the creation of several operators that are specifically designed around the rescheduling restrictions. This makes it more likely to find improvements in the local search step.

To not create infeasibilities, the neighbourhoods need to adhere to the directed arcs of the flow network. Because of the feasibility requirements, neighbourhoods might contain a limited number of solutions. Therefore, the steepest direction of descent (*best improvement*) is used with all operators in the VND. This requires calculating all solutions in the neighbourhood. While this can be time-consuming, possibly less time is spent calculating less promising solutions than using a first descent method, where a solution is directly accepted if any improvement is found. As a shake mechanism, a k-Destroy & Construct (D&C) operator is used. By destroying and reconstructing schedules, the algorithm prevents solution cycling and escapes local optima by changing the solution space.

The pseudocode of the developed GVNS can be found in algorithm 4. The algorithm runs while the maximum running time has not been reached and the number of continuous iterations without improvements has not reached the maximum (line 4). The shake happens in lines 7 and 8 using the D&C. k defines the number of schedules destroyed. The higher the value of k, the more destroyed schedules. The more destroyed schedules, the further the new solution space is from the current solution space. k loops over the set of k values in N_k (line 6). The set N_k contains decreasing values of k so that the algorithm diversifies first, and specifies afterwards. The D&C operator generates several new solutions and randomly selects one of the best to prevent solution cycling (line 9). Lines 10-13 show how the VND is used as a local search. The operators used in the VND are different from the D&C operator. The VND uses three local search neighbourhoods: *Swap*, *Replace* and *Insert*. The best solution from these neighbourhoods is selected as the current solution (line 17). All neighbourhoods will be explained in section 3.4.3.

Algorithm 4: GVNS

```
   Input: Original plan, Maximum time
   Output: Best optimized plan
 1 Initialize the best solution with the original plan:
 2 current_solution_value ← Cost of the original plan
 3 best_value ← current_solution_value
 4 while time < max_time and ImprovementLessIter < MaxIter do
 5     current_solution_value ← best_value;
 6     foreach k in $N_k$ do
 7         Destroy(k);
 8         Construct();
 9         Randomly select one of the best solutions;
10         Apply VND to selected solution;
11         foreach operator in (Swap, Replace, Insert) do
12             Calculate neighbourhood;
13             Save best neighbour;
14         if Selected neighbour is better than best_value then
15             Save the selected solution as the best solution:
16             best_value ← current_solution_value
17         Set the found solution as the current solution;
18     if current_solution_value ≤ best_value then
19         increment ImprovementLessIter ;
```

3.4.3 Operators

This section discusses the operators developed for the GVNS. There is a distinction between the main neighbourhood and the neighbourhoods used in the VND. The GVNS uses the $k-$Destroy & Construct (D&C) neighbourhood. The VND uses the Swap, *Replace* and *Insert* neighbourhoods. The pseudocodes of the neighbourhood operators can be found in appendix C.

Destroy & Construct To create schedules that are much different from the initial solution, a destroy heuristic is combined with a constructive heuristic. The destroy heuristic destroys k random driver schedules. This number k can be changed and is a parameter that can be tuned. Instead of choosing partial destroy, the complete driver schedules are destroyed. This is done to prevent reconstructing the schedules too similar to the initial schedule. The pseudocode of the destroy can be seen in appendix C, algorithm 6.

The destroyed schedules have to be repaired. A constructive algorithm is developed to repair the destroyed schedules. The pseudocode of the constructive algorithm is shown in appendix C, algorithm 7. The algorithm starts with finding the productive actions which lack a driver resource because of the destroy heuristic. This list of unassigned productive actions is sorted on their start time. Actions from this sorted list are iteratively assigned to driver schedules. The driver to which the action is assigned is determined pseudo-randomly using a probability based on their *suitability*. A driver's suitability is based on their required movement and idle time. Their movement time is the time required to reach the action's start location. Their idle time is the time between arrival at the location and the start time of the action. The weight w of the suitability factor can be tuned. Parameter tuning is described in chapter 5.1.

$$suitability\_score = w * move\_time + (1 - w) * idle\_time \tag{3}$$

Every action is assigned to one of the 3 most suitable drivers. This creates randomness in the construction of new shifts, which prevents reconstructing the same schedule. Driver schedules are considered full if the time remaining in their shift if they would go back to their crew base now, is smaller than the time buffer t. This time buffer is a parameter that can be tuned. If all productive actions are assigned, each driver is given a break. When no break is possible in a driver's schedule, the constructive heuristic resets and restarts. If, due to the randomness of the assignment of actions to driver schedules, not all productive actions are assigned, the constructive heuristic resets and starts again. If the solution cannot be repaired after several attempts, the original schedules will be reset and the destroy heuristic will be called again, destroying new schedules.

Swap The swap operator simply exchanges tasks between schedules where this is possible without introducing scheduling conflicts. A swap does not directly reduce driver movement time because a planned movement would be swapped to another driver's schedule. However, swapping might create redundant movements. Due to a swap, a driver might be planning to move from A to B and back to A. Such a movement has then become redundant. This operator can reduce driver movement time by removing redundant movements after a swap. The pseudocode of the swap heuristic is shown in appendix C, algorithm 8. The swap operator is visualised on a MCNF in figure 3.4.

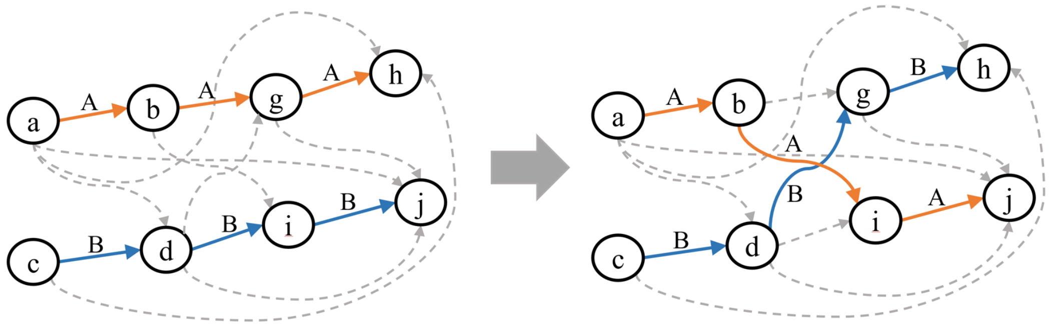

FIGURE 3.4: Swap Operator

Replace The replace heuristic finds productive action groups and tries to place them in another driver's schedule. Productive action groups consist of train rides and other productive actions. The replace operator tries to place the productive action groups over another driver's move action group. This might create a flow conflict in the new driver's schedule (planned to start the next action at location B, while the previous action ends at location A). New driver movements might have to be created to make sure all schedules remain feasible. When a driver's productive action group is removed from their schedule, move actions that were planned to move to the start location of the productive actions can become redundant. Such redundant move actions are removed, and new move actions can be generated to prevent flow conflicts. The pseudocode of the swap heuristic is shown in appendix C, algorithm 9. The replace operator is visualised on a MCNF in figure 3.5. In the figure, node *i* is no longer assigned after the replace. This can only occur when node *i* is not a productive action.

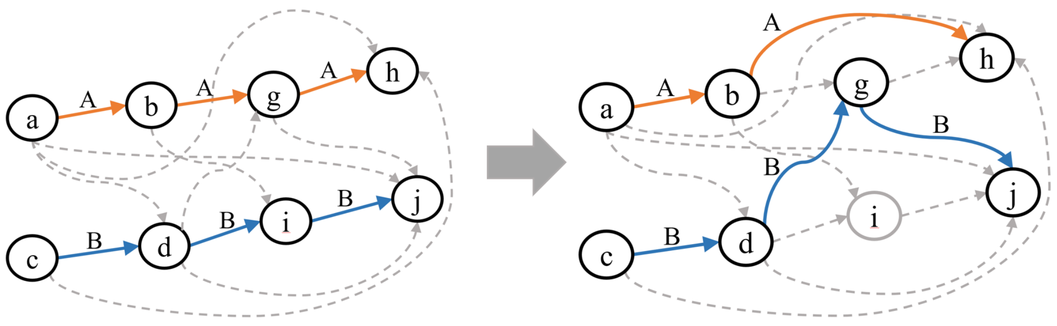

FIGURE 3.5: Replace Operator

Insert The Insert heuristic is similar to the replace operator. The difference is that the insert operator does not find move action groups but tries to insert productive action groups in another schedule. After insertion, redundant move actions are removed. To solve flow conflicts, new move actions can be generated. The pseudocode of the insert heuristic is shown in appendix C, algorithm 10. The insert operator is visualised on a MCNF in figure 3.6

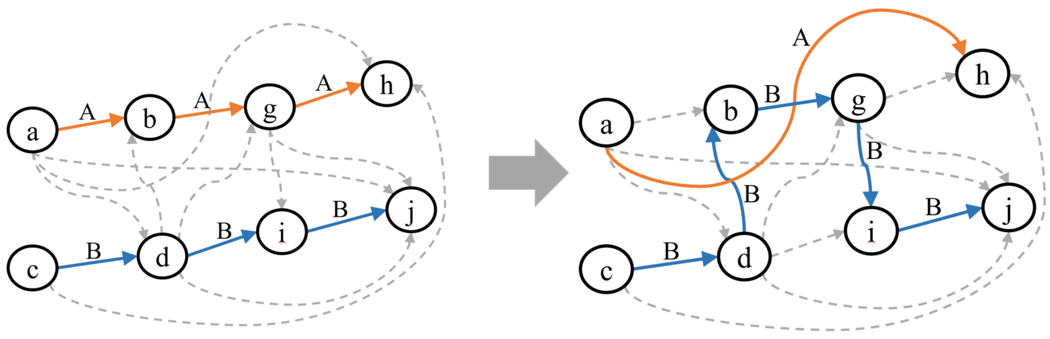

FIGURE 3.6: Insert Operator

3.5 Company Car Scheduling

Besides optimising the DP for driver movements, this research aims to determine the effect of company car scheduling strategies on company car utilisation. This is the second step of the methodology in table 3.1. In section 1.3.6 a company car scheduling strategy was suggested by the company. The company proposes to add company cars to routes that contain high taxi costs. This proposed strategy resulted from the company's analysis of taxi costs. Using the optimised DP, scheduling strategies can be compared to provide insights into their effect on company car utilisation. When comparing company car scheduling strategies, it is important to take the planning for the upcoming days into account. Individual (*optimised*) days can be sampled and linked together, representing consecutive days. By sampling and combining individual days, the long-term effect of a company car scheduling strategy can be determined. Every new day, the company car starts at the location where it ended the previous day. A First Come First Serve (FCFS) scheduling strategy is used as a performance baseline for the scheduling strategies. The FCFS strategy always uses a company car when it is available at a location, no matter the destination or future planning. The scheduling strategies that will be compared are:

0. **Performance baseline** First Come First Serve, always schedule a company car when available.
1. All company cars are restricted to high taxi-cost routes.
2. Some company cars are restricted to high taxi-cost routes.

The scheduling strategies require the analysis of routes that contain many driver movements after optimisation with the GVNS. The strategies should be compared based on the best subset of routes. By comparing these strategies, it can be determined if adding additional cars with route restrictions is beneficial compared to adding cars without route restrictions. Different route restrictions can be tried based on routes that contain the most driver movements.

3.6 Taxi Usage Evaluation

In the problem analysis (1.5) it was determined that the high number of driver movements in the DP is the core problem causing the taxi costs to exceed the budget. It was assumed that reducing driver movements and optimising company car scheduling would reduce taxi costs. The final step of the methodology from table 3.1 is to evaluate the combined effect of Day Plan optimisation and company car scheduling on taxi usage. The best company car scheduling strategy can be applied on both optimised and original DPs. A taxi usage baseline can be set using the FCFS company car scheduling strategy applied on the original schedules.

The combined effect of the DP optimisation and company car scheduling on taxi usage should be determined based on a longer horizon because taxi usage depends on company car usage. Company car usage depends on the planning horizon, as explained in section 3.5. The FCFS scheduling strategy can be applied to the optimised DPs to see if optimising for driver movements reduces taxi costs even if no changes are made to company car scheduling. The following combinations of Day Plans and company car scheduling methods will be compared:

0. **Performance baseline** Original DP + FCFS company car scheduling strategy.
1. Original DP + best company car scheduling strategy.
2. Optimised DP + FCFS company car scheduling strategy.
3. Optimised DP + best company car scheduling strategy.

3.7 Conclusion

Throughout this chapter, the research questions related to the methodology have been answered. This section will give an overview of the decisions made in this chapter.

The first section (3.1) of this chapter discussed the data available for this research and its implications on the solution design in section 3.2. Because the available data is anonymous, individual driver shifts on consecutive days cannot be linked. To prevent problems with rostering rules and labour laws, the rescheduling model cannot change the start and end times of shifts. It must also make sure that every driver ends their shift at their start location. All productive actions from the input DP have to be included in the optimised schedule, and every driver should have a break, for the schedule to be considered feasible (RQ2).

Because of the anonymity of the data, driver route and material knowledge cannot be included in the rescheduling model. Material knowledge is mostly similar for international drivers and can thus be removed without loss of generality. The removal of route knowledge does make the rescheduling slightly less restricted. Scheduling drivers to use public transport would require the implementation of the NS API in the rescheduling model. This is beyond the scope of this research. Driver movements have been simplified to taxis and company cars. The KPI driver movement percentage allows the results to be compared to the original DP (RQ3).

The mathematical model of the crew/rolling stock rescheduling problem from Sato et al. (2010) was adjusted to minimise costs based on planned taxi rides and taxi rides due to missing company car resources (RQ4). Instead of using a penalty cost for lacking drivers, it enforces that all productive tasks must be scheduled. A unique start and end node has been added for each driver resource that ensures that drivers end their shifts at their start location. The updated mathematical model is shown in model (2a) — (2k).

A three-step approach is taken to reduce driver movements. First, a GVNS is developed (algorithm 4) to optimise the Day Plan regarding driver movements (RQ5). Based on the optimised Day Plans with reduced driver movements, several company car scheduling strategies will be evaluated. Finally, the combined effect of Day Plan optimisation and company car scheduling on taxi usage will be determined.

The GVNS includes a k-Destroy & Construct (*D&C*) neighbourhood and a VND using the *swap*, *replace* and *insert* neighbourhoods. All neighbourhoods ensure schedule feasibility at all times to prevent the GVNS from introducing irreparable conflicts. The neighbourhoods in the VND use a best-improvement method. Section 3.5 discusses how the effect of company car scheduling strategies on company car utilisation can be evaluated (RQ6) and compared to a FCFS baseline. Two strategies to be tested are restricting company cars to routes with many driver movements and restricting a selection of company cars to specific routes that contain many driver movements. Section 3.6 discusses how the effect of both DP optimisation and company car scheduling on taxi costs will be determined.

The next chapter (4) discusses the problems with the available data and how these problems are solved. Chapter 5 discusses the experiments and results.

4 Data

Section 3.1 of the previous chapter already described the data available in this research. Section 4.1 describes the problems with the available data and how these problems are circumvented with a data generation method. Section 4.2 describes the data generation method and answers the research question 6 *What data should be used to test the model?* Section 4.3 answers research question 7 *What are the current Day Plan's KPIs?* and gives the KPIs of the generated datasets and compares them to the KPIs of the actual Day Plan.

4.1 Data accuracy

The company provided a year of historical data (2022) as described in section 3.1. The original driver schedules were recreated with the driver actions planned in the Day Plan. The actions are grouped based on the distinctive combination of shift number and shift date. Original driver schedules are generated by sorting the planned actions by their start time. An example schedule was given in figure 3.1. However, many original schedules do not look like the schedule in figure 3.1. The data provided by the company is distorted, which results in impossible original schedules.

There can be multiple actions assigned to a single driver shift at the same time, or there can be flow conflicts that make the driver schedules infeasible. When an action ends in a location and the next planned action starts at another location, there is a flow conflict. The recreated shifts do not all start and end in the same location, which they should for the shifts to be considered feasible. Some shifts last longer than what's allowed by the law, which shouldn't happen. Without going into detail about the specific actions planned, figure 4.1 shows an example of an infeasible shift from the data. This schedule contains all actions assigned to a single driver on that day. This schedule is infeasible because it contains up to 3 actions planned at the same time.

There are numerous possible reasons why the historical data is distorted. Firstly, the exact moment the DP is copied to the database is unknown. Because the DP is continuously changed and updated during the day, the data could be copied to the database while changes are being made. Another reason is the automatic adjustments of planned train actions due to changes in the timetable. When the infrastructure owner assigns a new time slot to a train, the corresponding train action in DBc's planning system automatically copies the new depart and arrival times. This changes the train task for the driver, but the driver schedule is not fixed automatically. This can cause a train action to overlap with another action in the driver's schedule. Sporadically, a driver's shift starts and ends in different places, which is not allowed. The reason for this is unknown. The large number of incorrect schedules in the data suggests that they may not accurately represent the actual Day Plan.

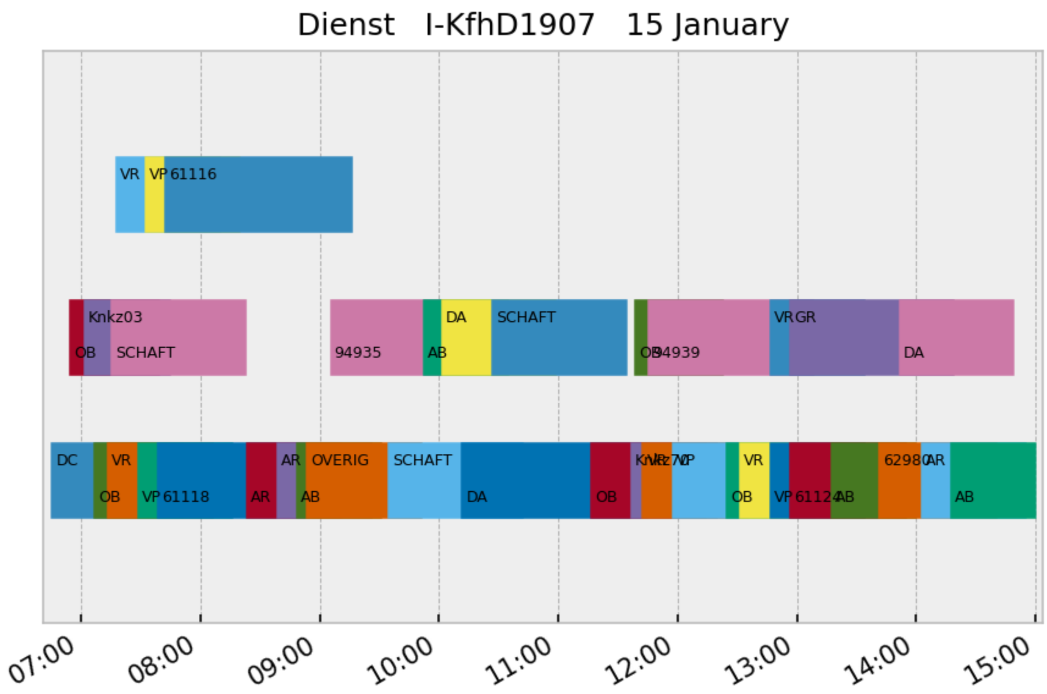

FIGURE 4.1: Example driver schedule on the 15 of January 2022

Because the rescheduling model uses the original DP as input, any conflicts or errors in the original data must be repaired. If schedules with flow conflicts are used as input to the model, the conflicts must be corrected before any optimisation can be performed. Making sure that drivers end their shift in their starting location is of high priority for DBc. Input shifts where the end location is not the same as the starting location should be repaired before they can be optimised. To avoid problems experienced with the provided data, we could sample with replacement, only from the correct schedules. Due to the quantity of incorrect schedules, this may potentially introduce bias into the model's input. Therefore, a method was developed to generate entirely new feasible schedules based on the original data. The schedule generation method is discussed in section 4.2.

4.2 Schedule Generation

A method is developed to generate new data. To ensure schedule feasibility, the method makes sure that every driver ends their shift on time in their starting location. It also schedules only logical flows. To make the generated data resemble the original DP, it uses parameters deduced from all original data, including the infeasible shifts. The parameters are:

Number of shifts Average number of shifts from a list of the number of shifts each day of the original data.

Probability of shift belonging to crew base Probability a shift starts at each crew base, based on all data disregarding differences between days.

Probability of start location per crew base Each crew base can have several starting locations, which is shown by the shift number. The probabilities for start locations of each crew base are based on all shifts.

Probability of actions Actions are drawn based on the probability of the categories train, productive and move actions, normalized to 1.

Probability of specific actions The original data contains various productive actions, of which the definition is less important for planning purposes. But because of their different durations, their probabilities are determined.

Average and standard deviation of action durations Most productive actions have varying times, and thus the average and standard deviation of the durations are determined.

Probability of train trips between locations When generating a train trip from a location, the probability of going to another location is based on the current location.

Durations of train trips Train trip durations are based on average durations for each location combination. While the actual duration is based on train characteristics, these are unknown and average times are suitable for planning purposes.

Probability of movements between locations A driver's probability of moving to a location is based on the driver's current location.

Duration of taxi rides Taxi times are predefined by the company for location combinations. Trips that have no predefined duration are assumed unlikely to occur and thus not considered in the shift generation.

Algorithm 5 shows the pseudocode of the data generation method. The algorithm creates individual schedules until enough shifts have been created. For every shift, it draws a crew base (line 2) and based on the crew base a start location (line 3). It generates actions based on the probability of action categories (line 5) such as movements and train rides. Destination locations are drawn with a probability based on the current location (line 5). Breaks are planned at random when this has not yet been done (line 9). The probability of planning a break increases the closer the current time is to the middle of the shift. The algorithm ensures drivers go back home with a probability based on the time required to go home and the time remaining in the shift (11).

Some parameters in the data generation algorithm can be adjusted to change the KPIs of the generated schedules. The two parameters w and $time\_buffer$ allow the KPIs of generated schedules to be influenced. For more idle time, the time between activities can be increased by increasing the wait time factor w. For a more productive schedule, w can be decreased. To create feasible schedules, drivers must go home at the end of their shift. The factor $time\_buffer$ determines how much time there has to be left in the shift for a driver to go back to their starting location.

Algorithm 5: Generate Schedules

```
Input: Number of schedules per day
Result: Generated schedules
foreach schedule do
    Draw a crew base;
    Draw a start location based on the crew base;
    repeat
        Generate actions based on probabilities;
        if a train movement is drawn then
            Optionally add other train actions such as shunting;
        After an action, add a random wait time, multiplied by the factor $w$;
        Draw a break if no break has been scheduled yet, based on
          probability and time remaining;
        Determine the time required to go home;
        Go home if time left in shift after going home $\leq time\_buffer$;
        if Driver went home then
            Driver shift full;
            Break;
        else
            Draw a new action;
    until schedule is full;
```

4.3 Performance baseline

To validate newly generated datasets, they can be compared to the actual DP KPIs. These KPIs are the percentage of shift time *idle/moving/train/productive*. These KPIs can be used to determine the similarity between the actual and the generated DP. 8 datasets have been generated that resemble the actual Day Plan based on the KPIs. The KPIs of the 8 generated datasets and the original DP KPIs are shown in figure 4.2. It shows that while there are slight differences, the KPIs are very similar. The similarity of these KPIs is important as these are expected to influence the rescheduling model's performance. A more productive DP means that there is less idle time that can be used to reschedule drivers. Testing the rescheduling model on DPs with similar KPIs shows the performance that can be expected if the rescheduling model is used on the actual DP. While the KPIs are similar based on the KPIs, they differ in the crew bases, actions and train rides because of the randomness in the data generation method. Only the number of shifts is the same. The number of shifts is set to the average number of shifts in the 2022 data. This ensures that the size of the datasets is a realistic representation of a day.

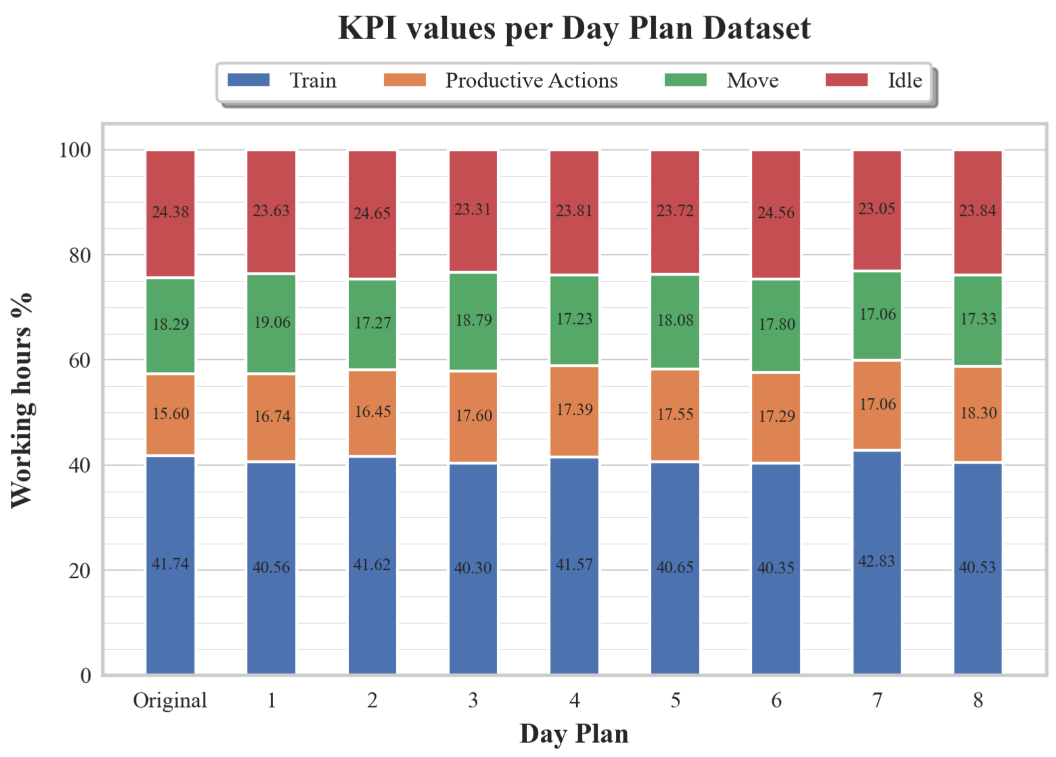

FIGURE 4.2: KPIs of generated datasets and the actual DP.

4.4 Conclusion

Throughout this chapter, the research questions 6 *What data should be used to test the model?* and 7 *What are the current Day Plan's KPIs?* have been answered. The used data is generated based on all international shifts in the original data. The instance size is determined by the average number of schedules in the original DP. This creates datasets of a realistic instance size. The KPIs of the generated datasets are similar to the actual DP so that the model performance can be compared. The KPIs of the original DP and the generated DPs can be seen in figure 4.2. The next chapter (5) describes the experiments that are performed.

5 Experiments and Results

This chapter discusses the experiments performed to answer research questions 8: *How much driver movement time can be reduced?*, 9: *How do different company car scheduling strategies affect company car usage?* and 10: *How is taxi usage affected by reduced driver movements and company car scheduling strategies?* Research questions 8 and 9 determine the individual effect of Day Plan optimisation and company car scheduling respectively. Research question 10 determines to what extent taxi costs can be reduced when they are combined.

The first section of this chapter (5.1) discusses the parameter tuning of the VNS and the parameters of the experiments. Section 5.2 discusses how the performance of the VNS is determined through experiments (RQ8). Section 5.3 discusses the experiments and results of the company car scheduling strategies (RQ9). Concluding, section 5.4 combines Day Plan optimisation and company car scheduling to determine the taxi usage reduction. All experiments are performed in Python 3.11.5 on an AMD Ryzen 7 5800H.

5.1 Parameter Tuning

Several parameters in the VNS can be tuned to optimise its performance. Experiments to optimise individual parameters are performed before the VNS is used to optimise the DPs. This section discusses these parameters and how they are set. This section also discusses the experimental parameters used in the company car scheduling experiments.

5.1.1 k — Number of schedules to destroy

The chosen k value in the k-Destroy & Construct operator determines the number of destroyed driver schedules. The more driver schedules are destroyed, the further the new solution space is from the original solution space. Because the D&C neighbourhood operator aims to shake up the algorithm, reaching a far-away solution space can be an advantage. However, destroying many schedules in a near-optimal situation could move the solution to a less promising solution space. The number of schedules destroyed by the destroy operator significantly affects the time required to create new schedules using the construct operator. This is shown in figure 5.1. From $k \geq 7$ a time, increase resembling exponential growth is recognised. Therefore, k is set no larger than 6. To allow the VNS to diversify and intensify, a range of k values is used. Based on figure 5.1, the chosen neighbourhood range is $N_k = [6, 5, 4, 3, 2]$. $k = 1$ is excluded, as destroying a single schedule can only result in the same repaired schedule. The VNS iterates over the k values from high to low. With these values, the VNS first diversifies, and intensifies with each new neighbourhood k.

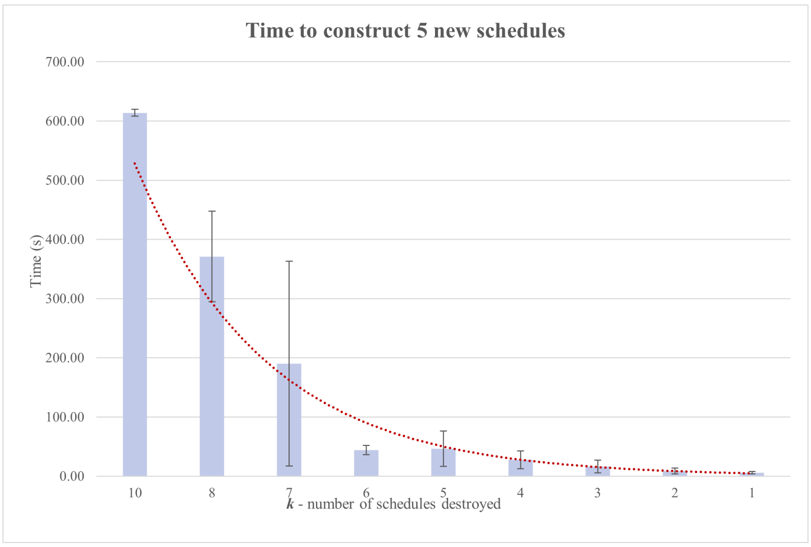

FIGURE 5.1: Parameter tuning of k, number of schedules to destroy

5.1.2 t - Time buffer

In the constructive heuristic, drivers' shifts are considered full if the time remaining in their shift, minus the time required to go home, is less than the time buffer t. The value of the time buffer t can affect the costs of the generated schedules, as well as the required time to create new schedules. A small t value can create problems by planning activities close to the end time of their shift. This might not leave enough time to go home before the end of their shift, causing the shift to become infeasible. A small t value might create better shifts because shifts are filled more. But it might take longer to create feasible schedules. A high t value might consider all drivers' schedules filled before all productive actions are scheduled, which would increase the time required to generate feasible schedules. The parameter t is compared (figure 5.2) based on the time required to generate 6 new schedules after destroying 5 driver shifts. The best of the 6 new schedules' total driver movement time is given in red on the secondary axis of figure 5.2. Because a time buffer t of 60 minutes performs best based on calculation time as well as driver movement time, the value of t is fixed to 60 minutes.

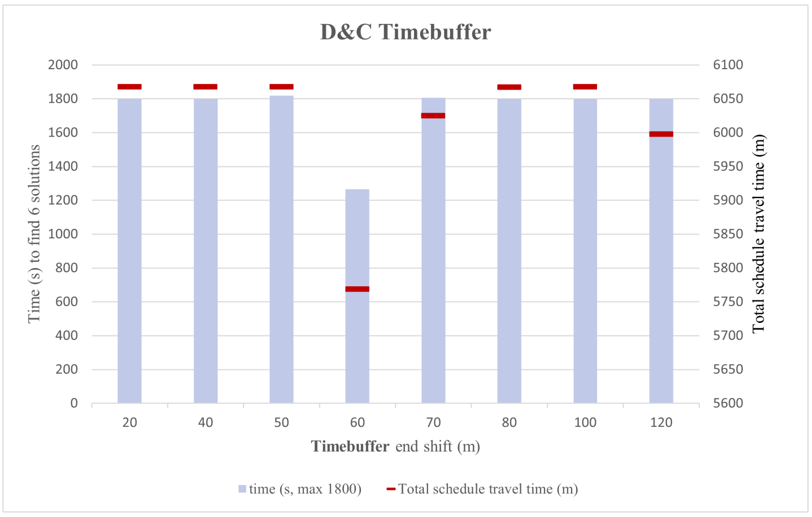

FIGURE 5.2: Parameter tuning of wait time to consider driver's shift as full

5.1.3 w - Move weight

The move weight factor determines the drivers' suitability score when assigning tasks in the construct operator. The score is based on $move\_time * w + (1 - w) * idle\_time$. A low score means a low combination of wait and move time. A higher move weight w penalises longer move times and reduces the impact of idle time on the decision of which driver to assign a task to. While idle time is 'free' in the cost calculation, too much idle time can create inefficient schedules. Different values of w are thus compared on total move time in generated schedules. The parameter values are compared by generating 50 new schedules after destroying 5 original shifts. The best and average total move time in the generated schedules are visualised in figure 5.3. It can be seen that 0.5 is the optimal move time weight factor based on both the average and best driver movement time. The driver move weight factor is thus fixed at 0.5.

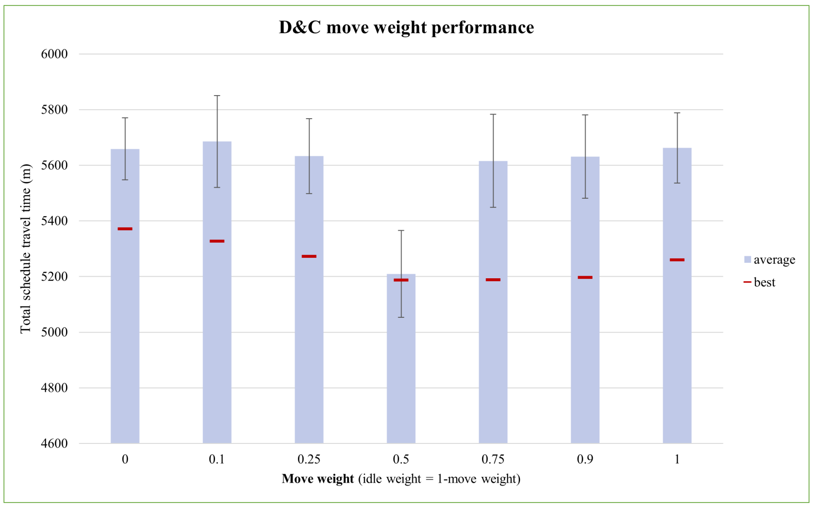

FIGURE 5.3: Parameter tuning of the move weight factor

5.1.4 Day plan optimisation

Before experiments with the VNS are performed, some experimental parameters are determined. These are shown in table 5.1. The running time is determined by evaluating the score progression over a long running time and taking a cut-off point where further movement time reduction is minimal. Based on figure D.1 in appendix D, the maximum running time is set to 150 minutes. Because of the relatively long running time, it was chosen to perform fewer replications on more DPs, instead of more replications on fewer different DPs. More DPs show how the VNS performs in different situations. Still, to increase confidence in the VNS performance, 3 replications are performed for each input DP.

parameter	**value**
Running time	150 minutes
Day Plans	8
Replications	3

TABLE 5.1: Parameters of the Day Plan optimisation experiments

5.1.5 Company car scheduling strategies

Some experimental parameters are also determined for the company car scheduling strategies experiments. Table 5.2 shows the parameters used for these experiments. For the FCFS scheduling strategy, it is expected that the result will after some time show a stabilising behaviour. At some point, most company cars likely end up in distant locations, but unless there are locations from which a driver never moves away, the system should not be terminating. Based on figure D.3 in appendix D, the period for the company car scheduling experiment is set to 20 days. From this moment, the system has stabilized. No warm-up period is used, as the initial effect of the scheduling strategy is as important as its long-term effect. Furthermore, it can be assumed that in a real situation, the company cars start at relatively good locations.

To simulate several continuous days, DPs are sampled with replacement from the 8 generated DPs. In each replication of the experiment, the same sequence of schedules is used for all scheduling strategies. This makes the strategies comparable per replication. Because the random sequence of DPs influences the company car usage, several replications with different DP sequences are required to determine the average performance of the scheduling strategies. The minimum number of replications for a certain confidence interval would differ for every scheduling strategy. Therefore, instead of calculating the minimum number of replications, it is determined visually based on figure D.2 in appendix D. From 10 replications, the graph is much smoother than with 5 replications, but 20 replications still have reasonable running time. The number of replications is thus set to 20.

parameter	**value**
Warm-up period	0
Days	20
Replications	20

TABLE 5.2: Parameters of the company car scheduling experiments

5.1.6 Taxi cost evaluation

The experiments to evaluate the combined effect of the DP optimisation and company car scheduling on taxi costs are similar to the experiments for the scheduling strategies. The performance has to be determined over a longer period because taxi usage depends on company car usage. In the scheduling strategies experiments, the short-term performance is important as it shows how the strategies perform when company cars start at ideal locations. In day-to-day operations, it is unlikely that company cars are in ideal locations. For evaluating taxi costs, the initial performance with company cars starting in ideal locations is thus less important.

The system's stable performance depends on company car utilisation. It was previously determined from figure D.3 that company car usage with the FCFS scheduling strategy stabilises after 20 days. Taxi usage is directly linked to company car usage: a 1-hour increase in company car usage means a 1-hour decrease in taxi usage. Figure D.4 in appendix D shows how much taxi time is used every day with an FCFS company car scheduling strategy using the original Day Plans. It shows that taxi usage initially increases and approximately stabilises at 20 days. This 20-day period corresponds with the stabilisation point in figure D.3 in appendix D. To evaluate both short-term and long-term taxi usage, the experiment horizon is increased from 20 to 50 days.

Figure D.4 also shows that the very first day has a large increase in taxi time as the company cars relocate from their ideal start locations. While this initial behaviour is important for evaluating company car scheduling strategies, it is unlikely to be experienced in reality. Therefore, a warm-up period of 1 day is used for the taxi cost evaluation experiments. This allows both short-term and long-term taxi usage to be evaluated but disregards the initial effect of company cars starting in ideal locations as this is unlikely to happen in reality. Because the taxi cost evaluation experiments determine the combined effect of Day Plan optimisation and company car scheduling on taxi usage, the replications have been increased from 20 to 100 to increase confidence in the results. Table 5.3 shows the parameters used for these experiments.

parameter	**value**
Warm-up period	1
Days	50
Replications	100

TABLE 5.3: Parameters of the taxi costs evaluation experiments

5.2 Day Plan Optimization

This section gives the result of the DP optimisation experiments and answers research question 8 *How much driver movement time can be reduced?* As described in chapter 3.2, all productive actions in the input DP must be scheduled in the optimised DP. Shift durations and thus the total amount of working hours cannot be changed. Therefore, the only KPI besides driver movement time that can change is the *idle time*. The idle time should increase relative to the found decrease in driver movement time. We are interested in the reduction of driver movement time in the different DPs.

The 8 different DPs are optimised using the VNS. This is repeated 3 times for every input DP. The average scores of each experiment are given in figure 5.4. The figure shows the decrease in total driver movement time over the running time of the algorithm. The total reduction in driver move time per DP is visualised in figure 5.5.

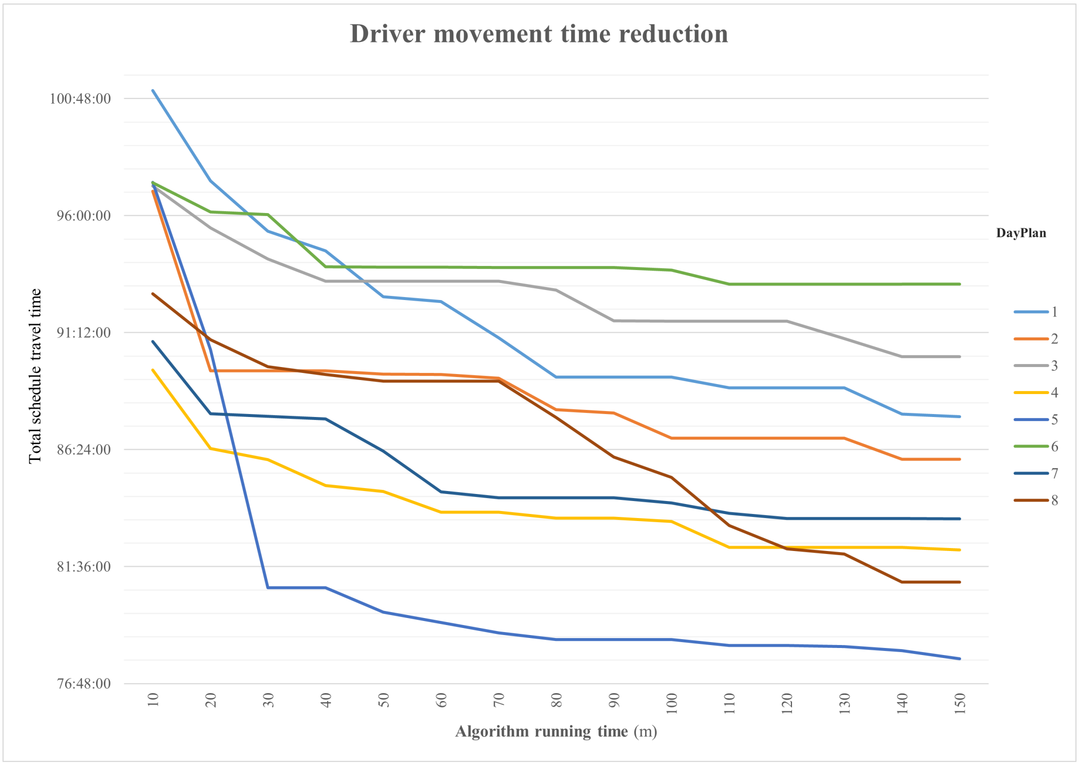

FIGURE 5.4: VNS results: Driver move time progression

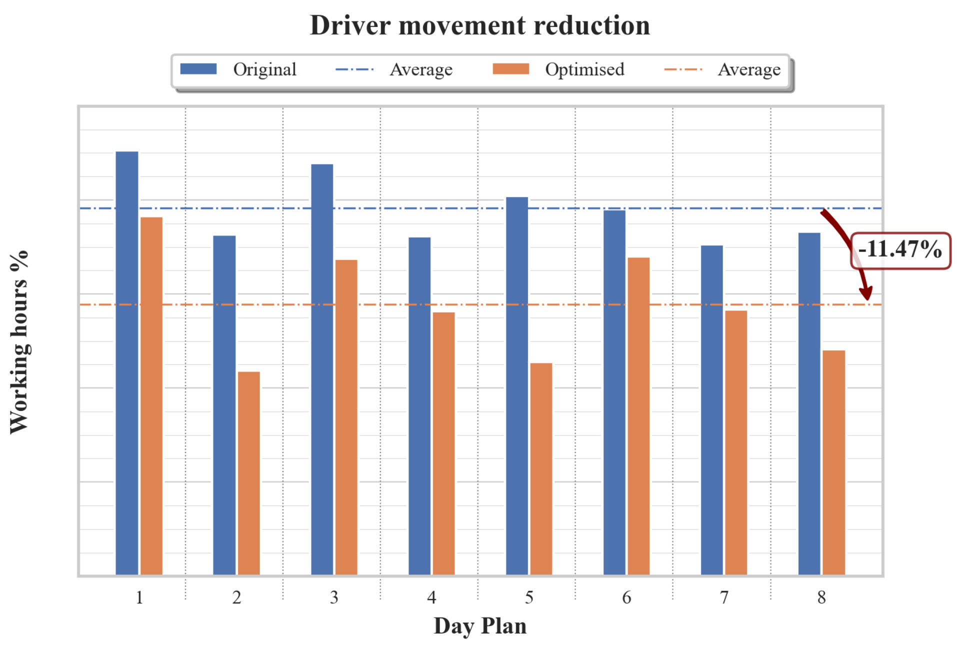

FIGURE 5.5: VNS results: Move time reduction per DP

Figure 5.5 shows that the difference between the average original movement time and the average optimised movement time is -11.47%. To give context to this value it must be compared to the original KPIs of the planning phases. Figure 5.6 shows the driver move time KPI for the original planning phases and shows what the result of an 11.47% reduction would be. The reduction would mean the driver movement time is reduced to below the Week Plan, the phase with the lowest percentage of driver movements of all scheduling phases. Table 5.4 gives the average and standard deviation of the reduction per DP. The standard deviation of the average reduction is 5.86%.

DP	**Average**	**St. dev**
1	-13.24%	0.81%
2	-11.33%	3.40%
3	-7.96%	2.12%
4	-8.23%	0.99%
5	-25.88%	2.16%
6	-5.48%	0.18%
7	-9.35%	1.70%
8	-12.76%	2.42%
Average	**-11.78%**	**1.72%**

TABLE 5.4: Driver movement decrease per Day Plan

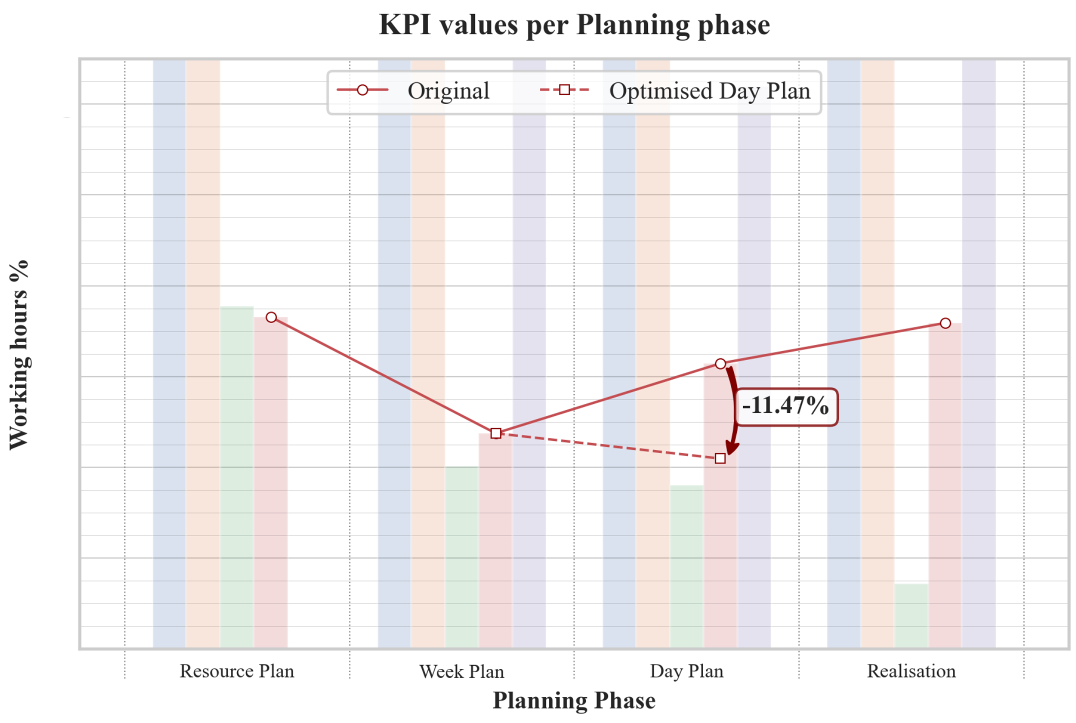

FIGURE 5.6: The effect of driver movement time reduction compared to the planning phases.

5.3 Company Car Scheduling

This section shows the result of the company car scheduling strategy experiments and answers research question 9 *How do different company car scheduling strategies affect company car usage?* The optimised DPs are used as input to these experiments. The same sequence of DPs is used for every scheduling strategy. The total driver movement time in the input DPs is thus the same for every scheduling strategy. We thus want to maximise company car usage. More company car usage means fewer taxi rides. The tested company car scheduling strategies are:

FCFS A baseline to compare performance, always uses company cars when available

TopLocations Restrict all company cars to only ride between locations that contain the most driver movements

CarRestrictions This strategy restricts some cars to only ride between locations with many driver movements. Cars are restricted when there are more than 1 car at a start location. All except the first car at each location are restricted.

CarRestrictions2 This strategy is a less restricted version of the CarRestrictions strategy. It allows restricted cars to visit all locations except for locations that are visited no more than twice a day.

Figure 5.7 shows how the strategies compare over a period of 20 days. Restricting all cars to only travel to the most visited locations (*TopLocations*) has the best long-term performance. This is logical, as cars never end up in less visited locations. The initial performance of this strategy is much worse. The performance in the first few days is important for company car scheduling. In the long term, there is a high chance that the schedule changes, which requires rescheduling. However, when rescheduling, it is important to have company cars in relatively useful positions. The long-term performance of a scheduling strategy should thus not be disregarded. The strategy *CarRestrictions* has a similar performance to FCFS but has a slightly better long-term performance. This strategy limits some cars to the most visited locations. Cars are restricted if there is more than one car in a start location. Cars 2, 3... are restricted, while the first car from a start location is never restricted. The strategy *CarRestrictions2* is a less restricted version of *CarRestrictions*, allowing the restricted company cars to visit all but the least visited locations (locations that are visited once or twice in the day). This strategy still shows a similar short-term performance as the FCFS strategy, but has a better long-term performance than *CarRestrictions*. It combines the short-term performance of the FCFS strategy with much better long-term performance.

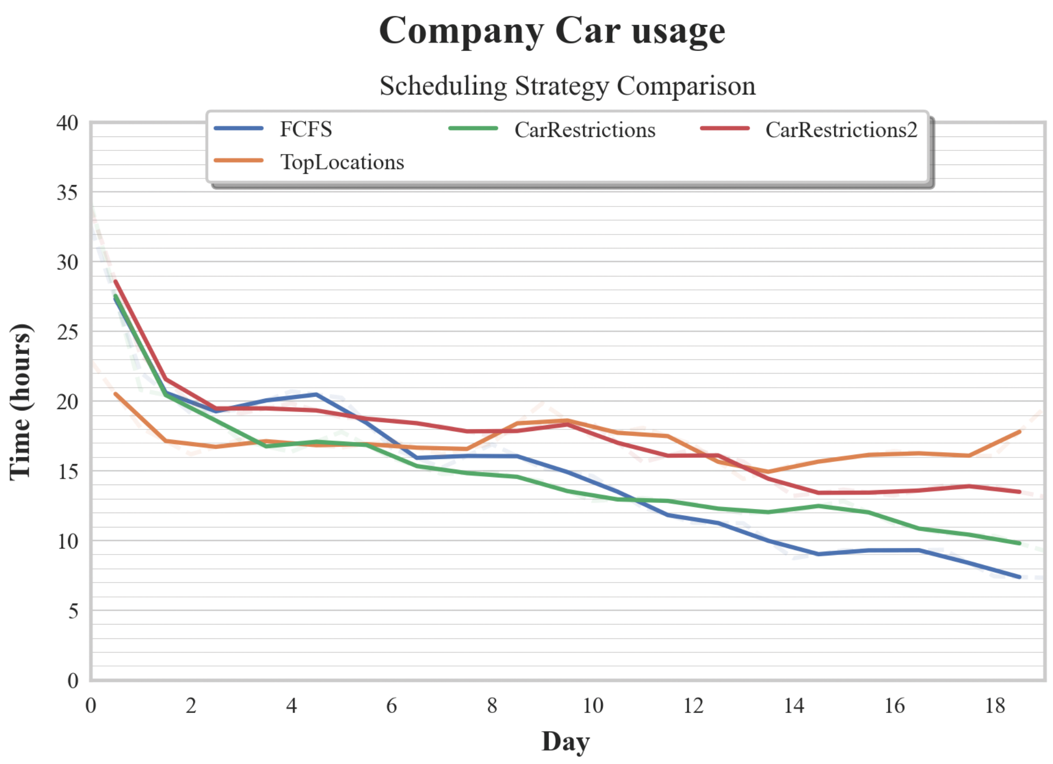

FIGURE 5.7: VNS results: Move time reduction per DP

5.4 Taxi Cost Evaluation

The experiments from section 5.3 resulted in a company car scheduling strategy that optimises company car utilisation by balancing short- and long-term performance. The short-term performance is comparable with the greedy FCFS strategy, but the long-term performance is much better. This last experiment determines the combined effect of the Day Plan optimisation and company car scheduling strategy on taxi usage. A taxi usage baseline is set with the FCFS company car scheduling strategy on the original Day Plans. Figure 5.8 shows the results of the taxi cost evaluation experiments. It shows the the taxi usage up to the stabilisation point (20 days). Figure D.5 in appendix D shows the stable long term taxi usage (20 - 50 days).

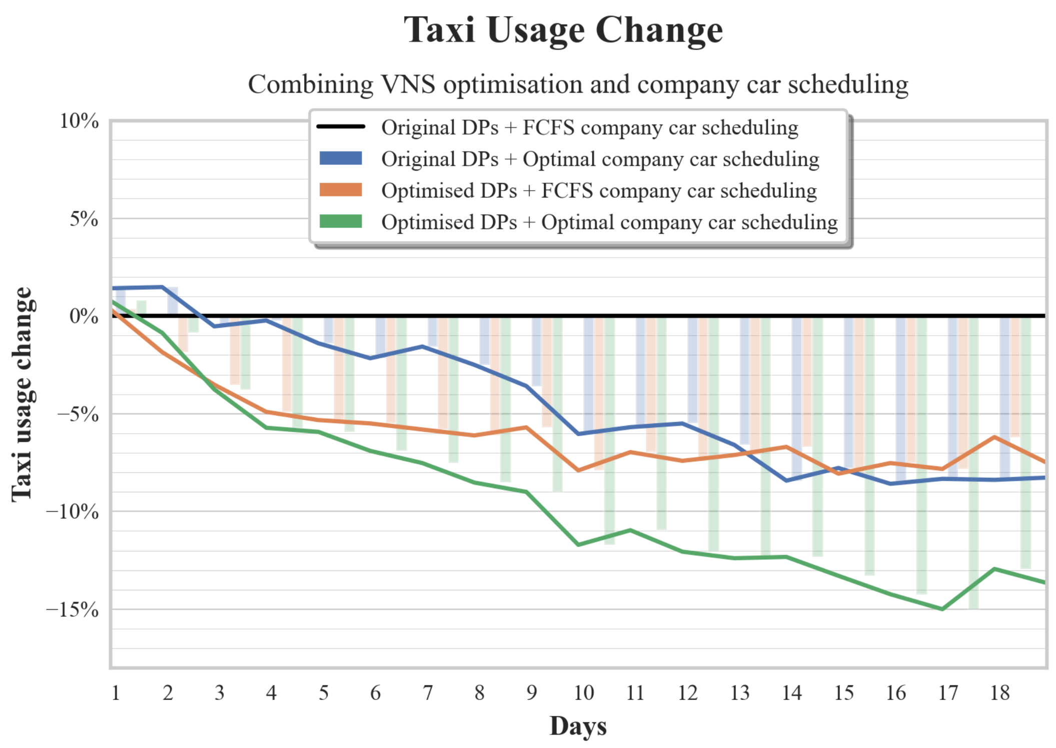

FIGURE 5.8: Taxi usage decrease with optimised Day Plans and company scheduling according to the best-found strategy in section 5.3

The **black** horizontal line represents the taxi usage with original DPs and the FCFS company car scheduling strategy. The performance of the other combinations are relative to this baseline. The **blue** line in figure 5.8 shows that using the optimised scheduling strategy on the original DPs slightly increases short-term taxi usage but decreases taxi usage after a few days. The **orange** line shows that the optimised DPs decreases taxi usage even if scheduled using the greedy FCFS company car scheduling strategy. The short-term performance of the FCFS scheduling strategy with optimised DPs is much better than either company car scheduling strategy with the original DPs. The long-term performance of the FCFS scheduling strategy with optimised DPs is slightly worse than the long-term performance of the optimised scheduling using original DPs. Both show approximately a 7.5% decrease in taxi usage at 20 days, the moment at which company car and taxi usage has stabilised. The stabilisation of the taxi usage can be seen in figure D.5 in appendix D. The **green** line shows that combining both DP optimisation and the optimised company car scheduling reduces taxi usage even more. At the stabilisation point of 20 days, a decrease of about 14% can be seen. The long-term decrease in taxi costs is slightly larger than the 11.49% decrease in driver movements realised with the VNS.

5.5 Conclusion

Throughout this chapter research questions 8, 9 and 10 have been answered. Together these research questions determine to what extent the research objective has been achieved. This section summarises the results of the experiments related to these research questions.

Section 5.2 evaluates the performance of the VNS used to optimise the DP by reducing driver movements. It answers research question 8: *How much driver movement time can be reduced?* Based on the average performance, the VNS can reduce driver movement time by 11.47%. This reduction would bring the total driver movements back to below the level of the Week Plan, which has the lowest percentage of driver movements of all planning phases.

Section 5.3 compares the effect of company car scheduling strategies on short- and long-term company car utilisation. It answers research question 9: *How do different company car scheduling strategies affect company car usage?* A baseline was set using a greedy FCFS strategy which maximises short-term company car utilisation with no regard for the future. The best strategy developed balances short- and long-term company car utilisation. This is done by limiting a subset of company cars to move only between locations that have several driver movements per day. The strategy shows similar short-term company car utilisation as the greedy FCFS strategy. The long-term performance is comparable to the strategy that focuses solely on long-term performance.

Section 5.4 evaluates how the Day Plans optimised with the VNS, together with the best company car scheduling strategy can reduce taxi usage. It answers research question 10: *How is taxi usage affected by reduced driver movements and company car scheduling strategies?* Figure 5.8 shows that reducing driver movements with the VNS reduces taxi usage both with the FCFS and optimal company car scheduling strategy. Using the optimal company car scheduling strategy, a long-term taxi usage decrease of $\sim$14% is achieved.

6 Conclusion and Recommendations

6.1 Conclusion

To conclude this thesis, this chapter will discuss the results and their implications. It will also give recommendations for the company regarding the implementation of the optimisation tool and company car scheduling. Research questions 11 *What is required for the optimisation model to be implemented?* and 12 *What can be recommended regarding company car scheduling?* will be answered. Finally, the limitations of the research and possibilities for further research are discussed.

In this research, a Variable Neighbourhood Search (VNS) is developed to optimise the Day Plan (DP) by reducing required driver movements to plan all productive actions. Experiments on 8 generated DPs with KPIs similar to the actual DP show that average driver movement time can be reduced by 11.47%. This reduction would reduce the driver movements to below the level of the Week Plan (WP). The WP and DP contain a similar amount of time spent driving trains, but the DP contains many more driver movements. Because the VNS can reduce the driver movements in the DP to below the level of the WP, it indicates that the inefficiency introduced between these planning phases can be reduced.

Company car scheduling strategies were compared on company car utilisation. The strategies were compared to a greedy First Come First Serve (FCFS) scheduling strategy. The best-found strategy balances short- and long-term company car utilisation. Experiments combining DP optimisation and company car scheduling show that reducing driver movements in the DP also decreases taxi usage. With the optimised DPs, long-term taxi usage decreases by $\sim$14%, when company cars are scheduled with the optimal company car scheduling strategy. When company cars are scheduled using the FCFS strategy, taxi usage decreases by $\sim$7.5% compared to the original DPs. The reduction in driver movement leads to an increase in idle time for drivers, as scheduled working hours are not adjusted. This increases train driver availability during operations. Idle train drivers could be used to further reduce taxi costs by moving company cars to locations where they are needed.

The results of the company car scheduling strategies experiments prove that it is important to take future planning into account when scheduling company cars. While this is something that DBc generally does, company cars are sometimes planned with no regard to the upcoming planning in Real Time Management. The company questions if adding company cars for specific routes that contain many taxi costs would be beneficial. The experiments show that adding additional company cars with restrictions to often visited locations is beneficial compared to adding these cars without restrictions. Restricting these cars to a small set of locations is less effective than limiting company cars to a larger subset of often visited locations. Because this research uses generated data, the experiments use routes with many driver movements determined from the optimised DPs. Would the company add cars for specific routes, then the routes should be determined from the actual taxi costs as well as planning data.

In conclusion, the developed VNS shows promising results for reducing driver movements in the Day Plan. The research into company car scheduling shows the importance of taking future planning into account when scheduling company cars. It also shows that adding company cars to specific routes can be beneficial, but the selection of routes should not be too small because it restricts company car use too much.

6.2 Recommendations

This thesis shows the potential for optimisation in the Day Plan planning phase. We recommend the company evaluate how the Day Plan can be optimised by either implementing the VNS developed in this thesis or by manual rescheduling. On a smaller scale at the harbour track, it was shown that by manual rescheduling, taxi costs can be reduced. This, combined with the potential improvement of the DP shown in this research, suggests that manual schedule optimisation might be a decent alternative to implementing automatic optimisation tools.

Besides optimising the DP, the company could evaluate how individual changes are added to the schedule in the period leading up to the DP. The increase in driver movements, as well as taxi costs, is the largest from WP to DP. Meanwhile, the time spent driving trains barely changes between these planning phases. This suggests that the individual changes added make the schedule less efficient. Logging of individual disruptions and changes is advised so that this process can be evaluated.

While the DP shows much potential for optimisation regarding driver movements, the real-time rescheduling process should not be disregarded. It is important to define how company cars should be planned in the real-time scheduling process and when or if taxis can be used. Company cars should not be planned without regard for their future planning. If the company would like to further evaluate the real-time re-scheduling process, it is essential to start tracking data about what changes and why. It is already known what routes have high taxi costs, but the reasons for these taxi costs are unknown. It would be helpful to know if there are certain trains that are often late or cancelled, or if there are other reasons these routes have many taxi rides. This requires more extensive data logging in the real-time management process.

6.3 Limitations

As described in chapter 3.2.2, the rescheduling model disregards driver-specific route knowledge because individual drivers cannot be related to original data. This relaxes the rescheduling model, which could cause the actual reduction in driver movements to be smaller. Before the rescheduling model can be implemented in practice, it would have to be adjusted to include this driver-specific route knowledge as well as material knowledge. The possibility of scheduling public transport is disregarded in this thesis because of the complexity of implementing the NS API. If the rescheduling model is to be implemented at DBc, public transport trips should be included. Using the NS API, a node in the MCNF can represent a complex public transport trip. The rescheduling model currently generates driver movements from a table of trip durations between all locations. The NS API could be used besides this table. The NS API can be used to generate new public transport trips, while the table can be used to schedule new taxi or company car rides. It would increase the calculation time of the VNS because for every new driver movement it has to be determined if public transport can be used.

While the datasets have been generated based on parameters derived from the original data, forcing the schedules to have KPIs similar to the actual DP changes the occurrences of certain actions. The similarity in action occurrences in the generated datasets and the original DP can no longer be guaranteed. To accurately evaluate the performance of the VNS, the similarity in KPIs was deemed more important than the occurrence of specific actions and routes. However, this causes the company car scheduling experiments not to reflect real company car usage. The datasets cannot be used to assess the benefit of adding company cars for specific routes. The general scheduling guidelines from the company car experiments are still valuable. Actual data about the routes with high taxi costs should be used to determine where company cars can be added for specific routes.

6.4 Possibilities for Future Research

The actual taxi usage is determined by both the crew schedules and company car schedules. Because the cost calculation in the mathematical model is based on both schedules, the minimisation of taxi costs is complex. Therefore, this research used a sequential approach in rescheduling the crew first and evaluating company car scheduling afterwards. The MCNF allows for representing both crew and company car schedules on the same network. The proposed mathematical formulation calculates the taxi costs based on the lack of company car resources. Future research could focus on developing an exact solution method to minimise taxi costs by simultaneous optimisation of crew and company car schedules. A method has to be developed that does not consider every possible taxi ride, as this makes the set of nodes in the MCNF too large for exact calculation. If a method does not require pre-calculation of every possible driver movement node, then the NS API can be included to generate individual public transport movements.

Bibliography

Abbink, Mobach, D., Fioole, P.-J., Kroon, L., Heijden, E., & Wijngaards, N. (2009). Actor-agent application for train driver rescheduling. *1*, 513–520. https://doi.org/10.1145/1558013.1558084

Abbink, Mobach, D. G. A., Fioole, P. J., Kroon, L. G., van der Heijden, E. H. T., & Wijngaards, N. J. E. (2010). Real-time train driver rescheduling by actor-agent techniques. *Public Transport*, *2*(3), 249–268. https://doi.org/10.1007/s12469-010-0033-6

ACM. (2021). Vervoersmonitor 2019 [[Online; accessed 09-January-2023]].

Borgonjon, T., & Maenhout, B. (2022). An exact approach for the personnel task rescheduling problem with task retiming. *European Journal of Operational Research*, *296*(2), 465–484. https://doi.org/10.1016/j.ejor.2021.03.047

Breugem, T., van Rossum, B., Dollevoet, T., & Huisman, D. (2022). A column generation approach for the integrated crew re-planning problem. *Omega*, *107*, 102555. https://doi.org/10.1016/j.omega.2021.102555

Budai, G., Maróti, G., Dekker, R., Huisman, D., & Kroon, L. (2010). Rescheduling in passenger railways: The rolling stock rebalancing problem. *Journal of Scheduling*, *13*(3), 281–297. https://doi.org/10.1007/s10951-009-0133-9

Cacchiani, V., Huisman, D., Kidd, M., Kroon, L., Toth, P., Veelenturf, L., & Wagenaar, J. (2014). An overview of recovery models and algorithms for real-time railway rescheduling. *Transportation Research Part B: Methodological*, *63*, 15–37. https://doi.org//10.1016/j.trb.2014.01.009

Cavone, G., Dotoli, M., Epicoco, N., & Seatzu, C. (2017). A decision making procedure for robust train rescheduling based on mixed integer linear programming and data envelopment analysis. *Applied Mathematical Modelling*, *52*, 255–273. https://doi.org//10.1016/j.apm.2017.07.030

Cavone, G., van den Boom, T., Blenkers, L., Dotoli, M., Seatzu, C., & De Schutter, B. (2022). An mpc-based rescheduling algorithm for disruptions and disturbances in large-scale railway networks [Accepted Author Manuscript]. *IEEE Transactions on Automation Science and Engineering*, *19*(1), 99–112. https://doi.org/10.1109/TASE.2020.3040940

Cullinane, K., & Toy, N. (2000). Identifying influential attributes in freight route/mode choice decisions: A content analysis. *Transportation Research Part E: Logistics and Transportation Review*, *36*(1), 41–53. https://doi.org//10.1016/S1366-5545(99)00016-2

DBC. (2023). Presentatie [[Online; accessed 09-January-2023]].

Dollevoet, T., Huisman, D., Kroon, L. G., Veelenturf, L. P., & Wagenaar, J. C. (2017). Application of an iterative framework for real-time railway rescheduling. *Computers Operations Research*, *78*, 203–217. https://doi.org//10.1016/j.cor.2016.08.011

Ernst, A., Jiang, H., Krishnamoorthy, M., Nott, H., & Sier, D. (2001a). Rail crew scheduling and rostering optimization algorithms. In S. Voß & J. R. Daduna (Eds.), *Computer-aided scheduling of public transport* (pp. 53–71). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-56423-9_4

Ernst, A., Jiang, H., Krishnamoorthy, M., Nott, H., & Sier, D. (2001b). An integrated optimization model for train crew management [Cited by: 39]. *Annals of Operations Research*, *108*(1-4), 211–224. https://doi.org/10.1023/A:1016019314196

Fang, W., Yang, S., & Yao, X. (2015). A survey on problem models and solution approaches to rescheduling in railway networks. *IEEE Transactions on Intelligent Transportation Systems*, *16*(6), 2997–3016. https://doi.org/10.1109/TITS.2015.2446985

Gu, H., Zhou, M., Liu, Y., Wang, H., & Dong, H. (2022). Multi-disciplinary cooperation rescheduling for high speed trains with considering the connection of rolling stock and crew. *2022 IEEE 17th International Conference on Control Automation (ICCA)*, 135–140. https://doi.org/10.1109/ICCA54724.2022.9831895

Hansen, P., Mladenovic, N., & Moreno-Pérez, J. (2010). Variable neighbourhood search: Methods and applications. *4OR*, *175*, 367–407. https://doi.org/10.1007/s10479-009-0657-6

Hoogervorst, R., Dollevoet, T., Maróti, G., & Huisman, D. (2021). A variable neighborhood search heuristic for rolling stock rescheduling. *EURO Journal on Transportation and Logistics*, *10*, 100032. https://doi.org//10.1016/j.ejtl.2021.100032

Huisman, D. (2007). A column generation approach for the rail crew re-scheduling problem. *European Journal of Operational Research*, *180*(1), 163–173. https://doi.org/10.1016/j.ejor.2006.04.026

Jordan, M. I., & Mitchell, T. M. (2015). Machine learning: Trends, perspectives, and prospects. *Science*, *349*(6245), 255–260.

Josyula, S. P., Törnquist Krasemann, J., & Lundberg, L. (2018). A parallel algorithm for train rescheduling. *Transportation Research Part C: Emerging Technologies*, *95*, 545–569. https://doi.org//10.1016/j.trc.2018.07.003

Liu, X., Zhou, M., Ma, J., Wang, B., Zhan, S., & Dong, H. (2022). A two-stage stochastic programming approach for timetable rescheduling with reassignment of rolling stock under uncertain disruptions. *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, 4349–4354. https://doi.org/10.1109/ITSC55140.2022.9921843

Lusby, R. M., Haahr, J. T., Larsen, J., & Pisinger, D. (2017). A branch-and-price algorithm for railway rolling stock rescheduling. *Transportation Research Part B: Methodological*, *99*, 228–250. https://doi.org//10.1016/j.trb.2017.03.003

Lusby, R. M., Larsen, J., & Bull, S. (2018). A survey on robustness in railway planning. *European Journal of Operational Research*, *266*(1), 1–15. https://doi.org//10.1016/j.ejor.2017.07.044

Maekawa, Y., Minakawa, T., & Tomiyama, T. (2020). An enumeration-based approach for flexible railway crew rescheduling in disruption management. *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, 1–6. https://doi.org/10.1109/ITSC45102.2020.9294205

Maenhout, B., & Vanhoucke, M. (2018). A perturbation matheuristic for the integrated personnel shift and task re-scheduling problem. *European Journal of Operational Research*, *269*(3), 806–823. https://doi.org//10.1016/j.ejor.2018.03.005

Malucelli, F., & Tresoldi, E. (2019). Delay and disruption management in local public transportation via real-time vehicle and crew re-scheduling: A case study. *Public Transport*, *11*(1), 1–25. https://doi.org/10.1007/s12469-019-00196-y

Mercier, A., & Soumis, F. (2007). An integrated aircraft routing, crew scheduling and flight retiming model. *Computers Operations Research*, *34*(8), 2251–2265. https://doi.org//10.1016/j.cor.2005.09.001

Mladenović, N., & Hansen, P. (1997a). Variable neighborhood search. *Computers Operations Research*, *24*(11), 1097–1100. https://doi.org//10.1016/S0305-0548(97)00031-2

Mladenović, N., & Hansen, P. (1997b). Variable neighborhood search. *Computers Operations Research*, *24*(11), 1097–1100. https://doi.org//10.1016/S0305-0548(97)00031-2

Muroi, Y., Nishi, T., & Inuiguchi, M. (2010). Improvement of column generation method for railway crew scheduling problems [Cited by: 6; All Open Access, Bronze Open Access]. *IEEJ Transactions on Electronics, Information and Systems*, *130*(2), 275–283. https://doi.org/10.1541/ieejeiss.130.275

Nielsen, L. K., Kroon, L., & Maróti, G. (2012). A rolling horizon approach for disruption management of railway rolling stock. *European Journal of Operational Research*, *220*(2), 496–509. https://doi.org//10.1016/j.ejor.2012.01.037

Ning, L., Li, Y., Zhou, M., Song, H., & Dong, H. (2019). A deep reinforcement learning approach to high-speed train timetable rescheduling under disturbances. *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, 3469–3474. https://doi.org/10.1109/ITSC.2019.8917180

Potthoff, D., Huisman, D., & Desaulniers, G. (2010). Column generation with dynamic duty selection for railway crew rescheduling. *Transportation Science*, *44*(4), 493–505. https://doi.org/10.1287/trsc.1100.0322

Rezanova, N. J., & Ryan, D. M. (2010). The train driver recovery problem—a set partitioning based model and solution method [Disruption Management]. *Computers Operations Research*, *37*(5), 845–856. https://doi.org//10.1016/j.cor.2009.03.023

Sato & Fukumura. (2012). Real-time freight locomotive rescheduling and uncovered train detection during disruption. *European Journal of Operational Research*, *221*(3), 636–648. https://doi.org//10.1016/j.ejor.2012.04.025

Sato & Fukumura, N. (2010). An algorithm for freight train driver rescheduling in disruption situations. *Quarterly Report of Rtri*, *51*, 72–76. https://doi.org/10.2219/rtriqr.51.72

Sato, Sakikawa, S., Morita, T., Ueki, N., & Murata, T. (2009). Crew and vehicle rescheduling based on a network flow model and its application to a railway train operation. *IAENG International Journal of Applied Mathematics*, *39*(3), 142–150.

Sato, Tomoe, T., Morita, T., & Murata, T. (2010). Lagrangian relaxation method for network flow modelled crew and vehicle rescheduling. *2010 2nd International Conference on Advanced Computer Control*, *1*, 403–408. https://doi.org/10.1109/ICACC.2010.5486971

Vaidyanathan, B., Jha, K. C., & Ahuja, R. K. (2007). Multicommodity network flow approach to the railroad crew-scheduling problem [Cited by: 45]. *IBM Journal of Research and Development*, *51*(3-4), 325–344. https://doi.org/10.1147/rd.513.0325

Veelenturf, L. P., Kidd, M. P., Cacchiani, V., Kroon, L. G., & Toth, P. (2016). A railway timetable rescheduling approach for handling large-scale disruptions. *Transportation Science*, *50*(3), 841–862. https://doi.org/10.1287/trsc.2015.0618

Veelenturf, L. P., Potthoff, D., Huisman, D., & Kroon, L. G. (2012). Railway crew rescheduling with retiming [Special issue on Optimization in Public Transport+ISTT2011]. *Transportation Research Part C: Emerging Technologies*, *20*(1), 95–110. https://doi.org//10.1016/j.trc.2010.09.008

Verhaegh, T., Huisman, D., Fioole, P.-J., & Vera, J. C. (2017). A heuristic for real-time crew rescheduling during small disruptions. *Public Transport*, *9*(1), 325–342. https://doi.org/10.1007/s12469-017-0155-1

Walker, C. G., Snowdon, J. N., & Ryan, D. M. (2005). Simultaneous disruption recovery of a train timetable and crew roster in real time. *Computers Operations Research*, *32*(8), 2077–2094. https://doi.org//10.1016/j.cor.2004.02.001

Wang, Y., Zhang, Z., Huisman, D., D'Ariano, A., & Zhang, J. (2022). A lagrangian relaxation approach based on a time-space-state network for railway crew scheduling. *Computers Industrial Engineering*, *172*, 108509. https://doi.org//10.1016/j.cie.2022.108509

Wohlin, C. (2014). Guidelines for snowballing in systematic literature studies and a replication in software engineering. *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering*. https://doi.org/10.1145/2601248.2601268

Yuan, J., Jones, D., & Nicholson, G. (2022). Flexible real-time railway crew rescheduling using depth-first search. *Journal of Rail Transport Planning Management*, *24*, 100353. https://doi.org//10.1016/j.jrtpm.2022.100353

Zeng, Z., Meng, L., & Hong, X. (2018). Integrated optimization of rolling stock and crew rescheduling for high speed railway. *2018 International Conference on Intelligent Rail Transportation (ICIRT)*, 1–5. https://doi.org/10.1109/ICIRT.2018.8641575

Zhang, Y., Wang, M., Wang, R., Li, Z., & Zhang, N. (2020). A freight train dispatching approach for handling perturbations: A real case for the longest heavy-haul railway in china. *Smart and Resilient Transportation*, *2*(2), 85–99. https://doi.org/10.1108/SRT-09-2020-0009

Zhu, Y., & Goverde, R. M. (2020). Dynamic and robust timetable rescheduling for uncertain railway disruptions [Best Papers of RailNorrköping 2019]. *Journal of Rail Transport Planning Management*, *15*, 100196. https://doi.org//10.1016/j.jrtpm.2020.100196

A Additional Files

A.1 Snowball Literature Review

The Excel file contains 6 sheets: *'Start Set'*, *'Iteration 1'*, *'Iteration 2'*, *'Model Comparison'*, *'Model Comparison Short'*, *'Separate search terms'*

filename: Snowball_Literature_Review.xlsx

B Literature Review

B.1 Review Method

A snowballing approach as described by Wohlin (2014) was used to perform a systematic literature review of articles on railway rescheduling. An initial search in Scopus with the search query: *real-time railway rescheduling*, gave *8330* articles. The search word *real-time* is included because rescheduling the Day Plan happens shortly before the execution of the plan, which makes it similar to real-time rescheduling. The large number of articles is a result of the generic search query, which was chosen because it is so far unknown on which part of the rescheduling process should be focused. Articles were added to the tentative start set based on title, abstract and the number of citations. Articles focused on optimising passenger service levels were excluded as this is irrelevant for freight transport. The initial selection resulted in a tentative start set of 22 articles. Of these, 11 articles were selected for the actual start set based on the title, author, year and abstract.

Every iteration consists of a new tentative inclusion list and a smaller actual inclusion list. On the articles in the inclusion list, forward and backward snowballing is performed to find articles for the next iteration's tentative inclusion list. Articles in the tentative inclusion list are further evaluated to determine if they should be included in the actual inclusion list. Of the hundreds of articles found, 119 have been added to tentative inclusion lists over all iterations. The final inclusion list consists of 31 articles over all iterations. Table 2.1 gives an overview of the included papers and compares them on planning subject, model type and railway or planning representation. This section divides real-time rescheduling models into the rescheduling of rolling stock (RS) and crews (C). Table 2.1 gives an overview of the selected papers. The literature review of the timetables (TT) topic can be found in appendix B.2. A supplementary file containing an overview of the (un)selected papers, can be found in A.1.

B.2 Timetable Rescheduling

Timetable rescheduling is often the first step in railway rescheduling. Rescheduling of Crew and Rolling stock often assumes a previously rescheduled timetable and uses this as input (Lusby et al. 2017; Potthoff et al. 2010). This makes timetable rescheduling an important part of railway rescheduling.

Railways are often represented on either a macroscopic view (large, stations and connections) or a mesoscopic view (detailed, block sections and tracks). Zhang et al. (2020) developed a model that specifically handles the dispatching of freight trains. They use a mesoscopic block-section representation of the railway network. Only retiming and re-ordering are used as dispatching measures. The train movements and relations between trains are described using an alternate graph network. Cavone et al. (2022) created a model that rescheduled on both mesoscopic and macroscopic levels. They developed a heuristic based on the predictive control model. Their solution is developed for potential use in Automatic Train Control. An interesting addition to traditional rescheduling is performed by Cavone et al. (2017), who use Data Envelopment Analysis to add a self-learning decision-making procedure based on their mixed integer linear program. They represent the railway with similar track segments. Another method using self-learning decision-making procedures is proposed by Ning et al. (2019), who trained a Deep-Q network using Deep Reinforcement Learning. Machine learning is an upcoming subject in operation research (Jordan & Mitchell, 2015), and railway operations are no exception. Their model determined running times, dwell times and departure sequences for train timetable rescheduling. They represent the railway with stations and their connections instead of using higher detail block sections. A reduction in average total delay has been found compared to an First Come First Serve (FCFS) heuristic.

Because the duration of disruptions is often unknown, Zhu and Goverde (2020) use a multi-stage stochastic model to reschedule the timetables while considering stochastic disruption duration. The rescheduling model uses an event activity network. Activities are directed arcs from one event to another. Several events are included such as running, dwelling, pass-through, headway and turning activities. Stochasticity is included in the length of disruptions, which are assumed to have a finite number of possible realizations. Their representation of the rail network contains individual tracks through and between stations.

To utilize modern computing power, Josyula et al. (2018) developed an algorithm which can use the processor's parallelization capabilities. They model the railway as block sections. The rescheduling is done using a depth-first search in a binary tree model, where nodes represent train conflicts. Each branch can be searched by a separate CPU thread.

Veelenturf et al. (2016) developed a model that aims to maximise the number of train services that can still be run in case of a large disruption. They included rolling stock constraints to increase the possibility of getting feasible rolling stock schedules. The timetable rescheduling approach is based on an event-activity network. They also incorporate the possibility of rerouting train services. While the inclusion of rolling stock constraints can increase the probability of finding feasible rolling stock schedules, this model does not fully integrate rolling stock planning.

B.3 Integrated Models

Because the rescheduling steps of timetables, rolling stock and crews depend on each other, improvements can be expected by integrating parts of the rescheduling process. These integrated models are often much more complex and difficult to solve. Since real-time rescheduling requires solutions in minutes, integrated models might not always be suitable for real-time rescheduling (Cacchiani et al. 2014).

Veelenturf et al. (2012) integrate crew rescheduling with timetable rescheduling by introducing copies of shifts with slightly adjusted times. To limit the increase in complexity, only a subset of tasks are copied and adjusted. The model is solved with a depth-first search in a branch-and-bound tree with column generation in every node.

Liu et al. (2022) combine timetable rescheduling with the reassignment of rolling stock. Besides this integration, they also include uncertainty in disruptions. The problem is modelled as an event-activity network. It is solved as a two-stage stochastic program, which allows the inclusion of several possible disruption durations.

It is hard to create an integrated rescheduling model that reschedules both timetables, rolling stock and crews. To circumvent this issue but still reschedule everything together, Dollevoet et al. (2017) combined all rescheduling parts in an iterative framework.

Article	**Problem**	**Model**	**Railway representation**	**Subject**	**Solving method**
Zhu and Goverde, 2020	2 stage stochastic model	2 Stage Stochastic Model	arrival, departure events	TT	2 stage stochastic
Josyula et al. 2018	Train Timetable Rescheduling (TTR)	Binary tree model	Track sections	TT	depth-first search heuristic vs parallel algorithm
Zhang et al. 2020	Train Dispatching	Alternate Graph Network	macroscopic, block sections	TT	CPLEX
Cavone et al. 2017	Train Rescheduling	Data Envelopment Analysis (DAE)	Tracks, segments, stations	TT	Heuristic based on job-shop
Ning et al. 2019	Train Timetable Rescheduling (TTR)	Deep Reinforcement learning (DRL)	block sections, arrival and departure times	TT	Deep learning, NN
Walker et al. 2005	Combined train/driver scheduling model	Set Covering	Piece of work	TT + C	branch and bound with column and constraint generation
Rezanova and Ryan, 2010	Train Driver Recovery Problem (TDRP)	Set Partitioning	Train tasks, Driver Duties	TT + C	column generation, limited subsequence strategy
Veelenturf et al. 2012	operational crew rescheduling problem with retiming (OCRSPT)	Integer Linear Programm	Tasks	TT + C	Column Generation, Langrarian relaxation, Iterative neighborhood exploration
Maenhout and Vanhoucke, 2018	Integrated personnel shift and task re-scheduling problem (IPSTrSP)	Integer Linear Programm	Tasks	TT + C	Perturbation heuristic
Borgonjon and Maenhout, 2022	Personnel task scheduling problem with retiming (PTrSP-T)	Integer Linear Programm	Tasks	TT + C	Branch & Price
Liu et al. 2022	Event-Activity network	Two Stage Stochastic	two-stage stochastic programming	TT + RS	Exact, CPLEX
Cavone et al. 2022	Train Rescheduling	Model Predictive Control (MPC)	Large scale, macroscopic and mesoscopic models	TT + RS	bi-level MILP heuristic
Veelenturf et al. 2016	Train Timetable Rescheduling (TTR)	Event Activity Network	Macroscopic	TT + RS	CPLEX
Mercier and Soumis, 2007	Integrated aircraft routing, crew scheduling and flight retiming model	Integer Linear Programm	Mixed Integer Program	TT + RS + C	Benders decomposition
Gu et al. 2022	train cooperation rescheduling problem	Event Activity Network	Trains, stations, events	TT + RS + C	CPLEX
Dollevoet et al. 2017	Framework for railway rescheduling	Framework	Stations	TT + RS + C	Several

TABLE B.1: Overview of real-time railway rescheduling models.

C Pseudo Code

C.1 VNS neighbhourhoods

Algorithm 6: Destroy

```
Data: List of drivers and their shifts
Result: Modified shifts after random sampling
1 Randomly sample a driver from the list;
2 foreach node in driver's shift do
3     Remove the node from the shift;
4 Re-add the start and end node of the driver's shift;
```

Algorithm 7: Construct

```
Data: List of nodes, List of drivers and their schedules
Result: Assignment of nodes to drivers
1  while len(solutions) <= solutions_wanted do
2      Find all nodes not assigned to a driver's schedule;
3      Sort the nodes based on their start time;
4      foreach unassigned node in sorted nodes do
5          foreach available driver do
6              Calculate movement time to reach the start location of the node;
7              Calculate idle time;
8              Rank the driver based on the weighted average of movement and
                 idle time;
9          Randomly select a driver to assign the node with a probability based
             on the driver's weighted average;
10         Determine assigned driver's time to home;
11         if time left after going home < time_buffer_end then
12             Driver to home and shift full;
13     foreach driver in repaired drivers do
14         Add break to schedule if possible;
15     if All repaired drivers have break then
16         Add solution to the list of solutions;
17     else
18         Retry construct;
```

Algorithm 8: Swap operator

```
Data: Driver schedules
Result: Optimized schedule
1  foreach driver in drivers do
2      foreach action in driver's schedule do
3          Select partial schedule for the current action;
4          foreach otherDriver in otherDriver do
5              foreach otherAction in otherDriver's schedule do
6                  Select the other partial schedule;
7                  if Swap is feasible then
8                      Swap the actions between the two drivers;
9                      Remove unnecessary driver movements;
10                     if New schedule has lower costs then
11                         Save the new solution;
12                     Undo the swap (backtrack);
```

Algorithm 9: Replace operator

```
Data: Driver schedules
Result: Optimized schedule
1  foreach driver in drivers do
2      Group assigned actions based on their type (productive / move);
3      foreach productive action group in the driver's schedule do
4          foreach otherDriver in other drivers do
5              Group otherDriver's actions based on their type (productive /
                 move);
6              foreach action group of otherDriver do
7                  if Replace move group by the action group fits with time then
8                      Replace move group by the action group;
9                      Repair the schedule by adding new driver movements;
10                     Calculate cost of the new schedule;
11                     if Costs are decreased then
12                         Save the new solution;
13                     else
14                         Reset to the original schedule;
```

Algorithm 10: Insert operator

```
Data: Driver schedules
Result: Optimized schedule
foreach driver in drivers do
    Group assigned actions based on their type (productive / move);
    foreach productive action group in the driver's schedule do
        foreach otherDriver in other drivers do
            Group otherDriver's actions based on their type (productive /
              move);
            foreach action group of otherDriver do
                if Replace move group by the action group fits with time then
                    Replace move group by the action group;
                    Repair the schedule by adding new driver movements;
                    Calculate cost of the new schedule;
                    if Costs are decreased then
                        Save the new solution;
                    else
                        Reset to the original schedule;
```

D Additional Figures

D.1 Variable Neighbourhood Search

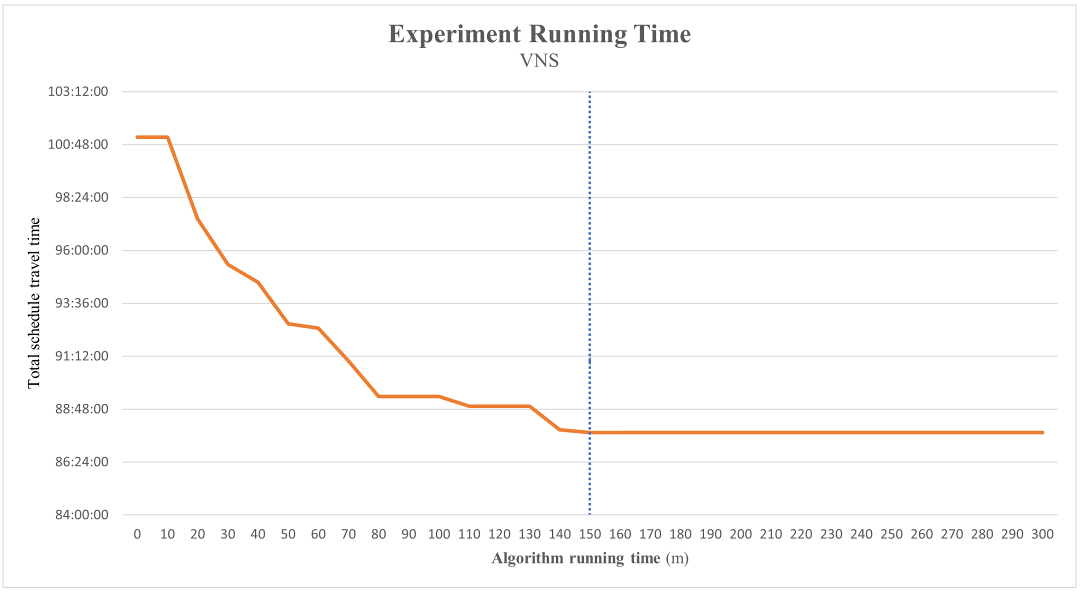

FIGURE D.1: Running time of the VNS experiments

D.2 Company Car Scheduling

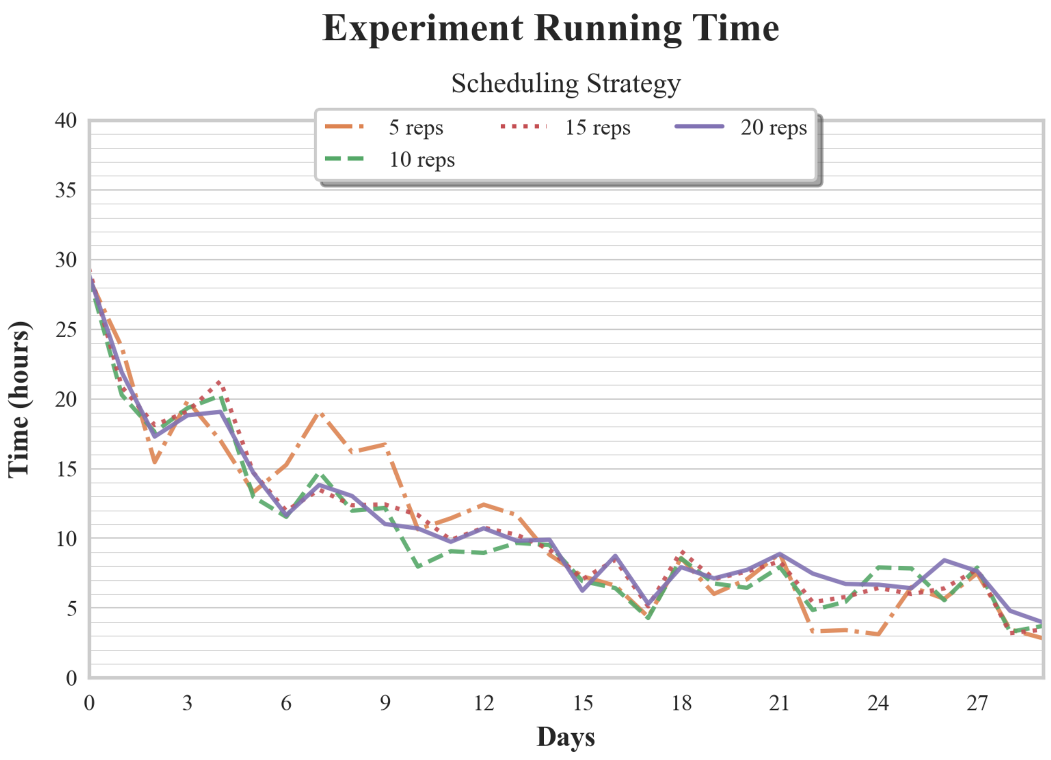

FIGURE D.2: Comparison of average results per number of replications

D.3 Company Car Scheduling

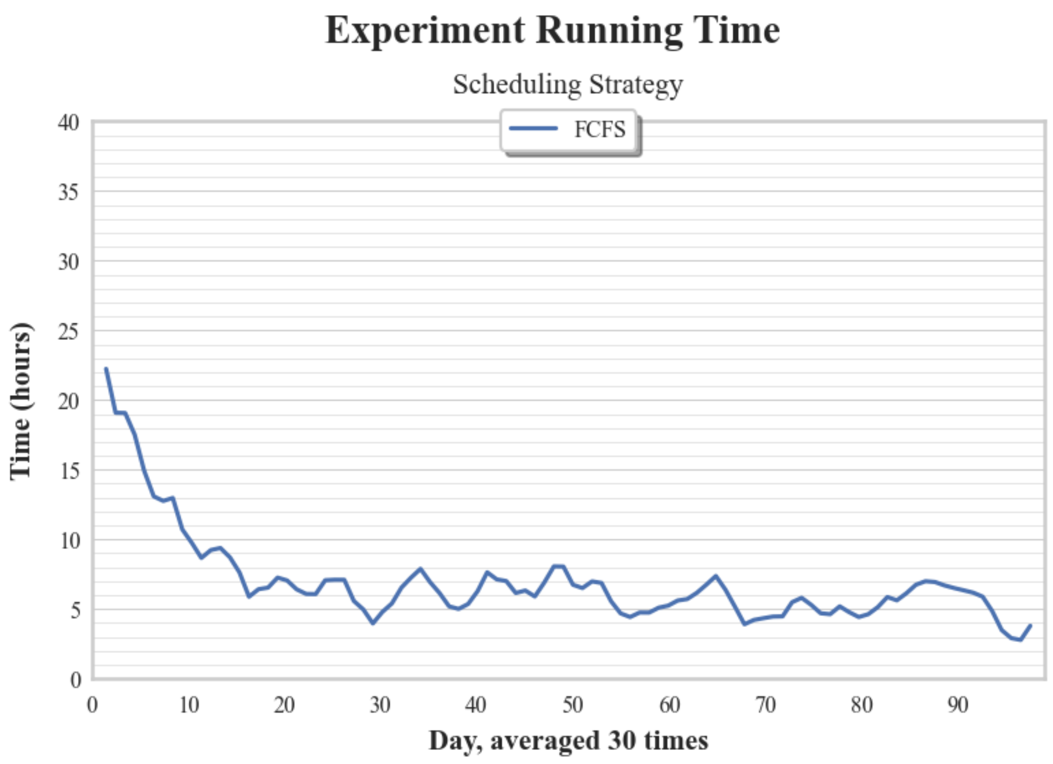

FIGURE D.3: Running time of the company car scheduling experiments

D.4 Taxi Cost Evaluation

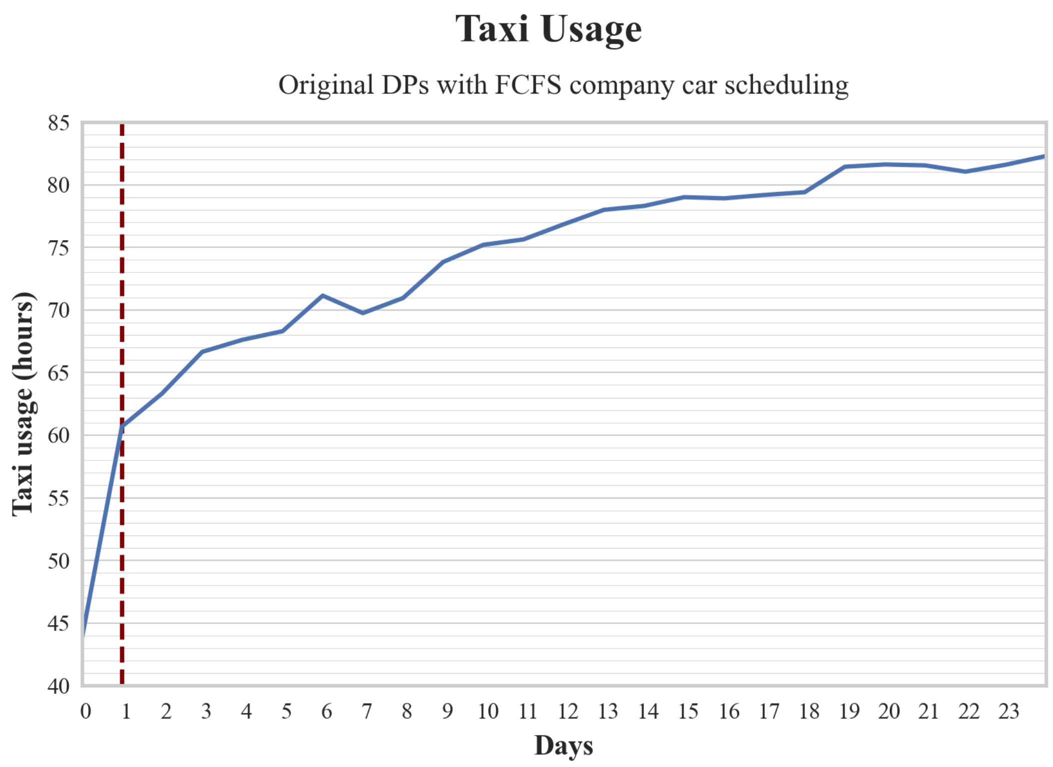

FIGURE D.4: Daily taxi usage over time

D.5 Long term Taxi Cost Evaluation

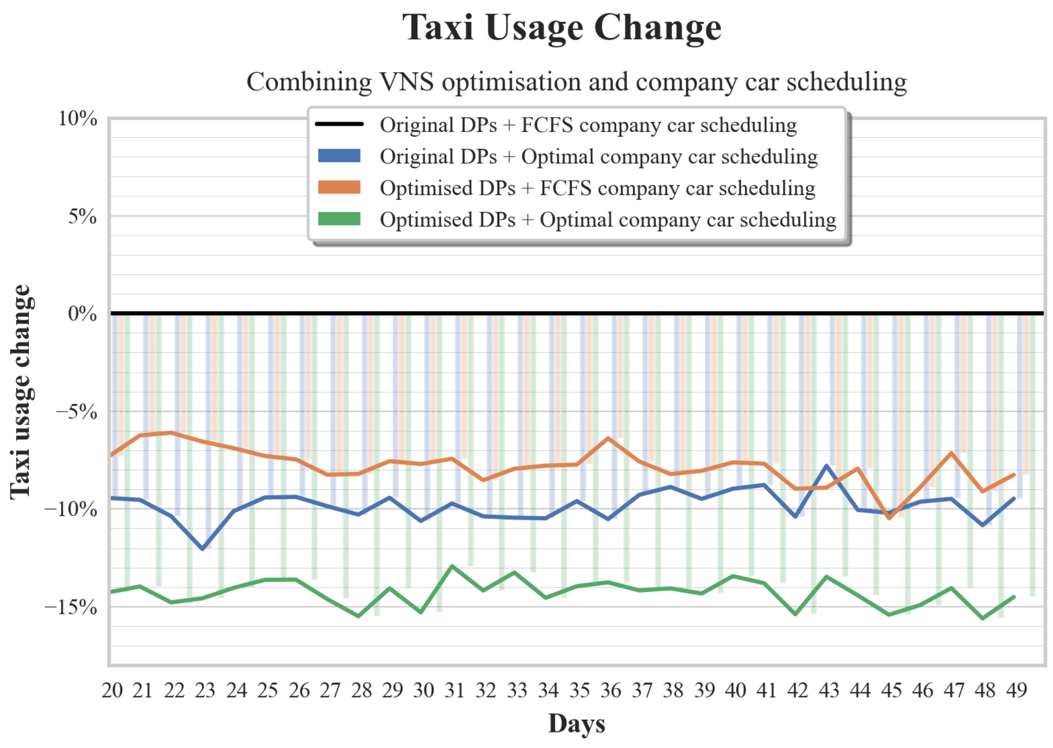

FIGURE D.5: Stable behaviour of daily taxi usage over time