UNIVERSITY OF TWENTE.

Planning first-line services on NS service stations

an exact approach

M.Sc. Thesis

Evelien H.R. Huizingh

(s1364472)

May 9, 2018

Examination Committee
dr. ir. J.M.J. Schutten (First supervisor)
dr. ir. L.L.M. van der Wegen (Second supervisor)
Ir. B. Huisman (Company supervisor)

Preface

Dear reader,

In the past nine months I wrote this thesis. This thesis is the final assignment that needs to be fulfilled to graduate the master Industrial Engineering and Management with the track Production and Logistics Management at the University of Twente.

First, I thank my supervisor Marco Schutten from the University of Twente for his guidedance to help me succesfully complete this project. The meetings we had every few weeks always gave me more than enough tips and starting points to continue my research. Second, I thank my second supervisor Leo van der Wegen of the University of Twente for also critically reading report.

This research project is part of a bigger research conducted by different teams at NS Techniek, a department within the NS Group. I am grateful for NS Group for providing me the opportunity to do a research internship at the Maintenance Development department. I thank my company supervisor Bob Huisman for taking me on in his team and giving me the opportunity to conduct my master thesis at his department. Finally, I thank my colleagues, fellow interns, and Gerjanne Dekker for their input, feedback, and help on my project during this time, and of course for the nice lunches every day.

I hope that you enjoy reading this thesis and that it is helpful for future NS projects or other research projects.

- Evelien Huizingh, May 2018

Summary

This thesis develops a model to plan first-line services at a service station of NS group (NS), the largest Dutch railroad company. First-line services include inspections, internal and external cleaning of trains, and small maintenance tasks. Currently, this plan is created manually but NS wants to automate this to support their planners. NS is expanding its fleet, and it is already difficult to find good first-line services plans. This becomes even more difficult in the future, hence this research.

Planning the first-line services is a subproblem of the service station planning problem that also includes the routing of trains, the (de)coupling of train units, the parking, and the personnel planning. However, we focus solely on finding a first-line services plan and leave the other subproblems out of scope.

The model is an Integer Linear Programming (ILP) model, solved in AIMMS Optimization software with the Cplex 12.8 solver. At service station *Kleine Binckhorst* the internal cleaning machines are regarded as the bottleneck of the first-line services plan. The model can find optimal plans up 16 train units that require three to six first-line services per job within a few minutes. All train units include the internal cleaning first-line service, which is regarded to be bottleneck at SB *Kleine Binckhorst*. In these plans the jobs are completed before their due date, and within a time window of one day. Adding more first-line services to jobs decreases number the problem instances for which a plan can be found without tardiness. Tardiness in a first-line services plan is caused by tight release and due dates of train units, and waiting time caused by the sequence in which the operations of train units are processed.

For future research, we recommend NS to continue to explore the exact method for solving the service station planning problem. The model can be integrated with other subproblems and extended such that it can be applied to service stations with another layout.

Abbreviations and definitions

Table 1: Abbreviations and definitions (Part I)

Abbreviation / term	Definition / Explanation
Carousel layout	Service station layout at which (most) tracks are free tracks, meaning that a train unit can enter and depart the track at both sides, creating a carousel route over the SB.
First-line services	Services provided at a service station
Flexible flow shop problem	An extended variant of the classical flow shop problem. A scheduling problem in which groups of identical machines are classified into stages and all jobs need to be processed in the same order.
HIP	An algorithm for solving the service station planning problem, based on heuristics (Hybride Integrale Planmethode).
Jobs	The train units on the service station
Machines	The tracks on the service station
ILP	Integer Lineair Programming, the model form of how the model in this thesis is defined.
Material number	Unique number of a train unit.
NSR	The department concerning the timetable for passenger transport (NS Reizigers)
OB	Largest, and most extensive maintenance station (Onderhoudsbedrijf)
Operations	The operations are planned in the model and consist of the processing times of one or more fist-line services.
Planner	There are two types of planners (werkvoorbereiders). One is the Logistics-Planner who is concerned with assigning tracks to a train unit, and the other, the Task-Planner who is concerned with assigning tasks to train units, and personnel to the tasks.
SB	Service station on which the first-line services are executed. (Service Bedrijf)

Table 1: Abbreviations and definitions (Part II)

Abbreviation/term	Definition/Explanation
Service station planning problem	The whole planning problem that NS needs to solve and consists of different subproblems. One of the subproblems is the first-line services planning problem that we solve in this thesis
Shuffleboard layout	Service station layout at which most tracks are LIFO (Last In First Out). Train units may only enter and depart the tracks from one side.
TC	Maintenance station, a little more extensive than a service station (Technische Centrum) but less than an OB.
TDL	Train operator (Treindienstleider). The TDL determines if it is safe for a train unit to drive.
TUSP	Train Unit Shunting Problem

Contents

1 | Introduction

An increasing number of people use train transport daily. NS Group, in Dutch often abbreviated and used in this report as *NS*, is the largest Dutch railroad company. NS is responsible for transporting more than 1.1 million passengers per day (Nederlandse Spoorwegen, 2016). To maintain a reliable and customer satisfying service, trains need to perform well and need to be clean. To do that, the trains need to be inspected, cleaned and minor maintenance tasks need to be executed often. This happens during night hours, when trains are not needed for passenger transport. In this thesis, the inspection, the cleaning, and the minor maintenance tasks are referred to as *first-line services*. This thesis focuses on the planning of the first-line services.

This chapter is structured as follows: Section 1.1 briefly introduces NS and provides the problem statement. Section 1.2 elaborates on the research goal and states the research questions.

1.1 Background NS and problem statement

Train transport was first possible in the Netherlands in 1839 (Nederlandse Spoorwegen, 2017a). Since then, several companies were responsible for the Dutch train transport. In 1937, almost a century after the introduction of the train in the Netherlands, NS was established (Nederlandse Spoorwegen, 2017b). The number of passengers that use train transport have increased immensely over the last decades. Currently, a few different companies provide train transport in the Netherlands. Daily, together the companies transport almost 2 million passengers over 3000 kilometers railway network in the Netherlands (Kroon et al., 2008), resulting in the Dutch railroads being among the busiest railroad networks in the world (Gestrelius et al., 2017).

NS is the backbone of the Dutch public transport by providing transport from a door-to-door perspective (Nederlandse Spoorwegen, 2016). NS is a service organization with a clear emphasis on passengers. The NS slogan is ‘passengers on first, second and third place’. To implement this vision, NS has three core focus points: a good train journey, transport from door to door, and world leading train stations. Within the core focus ‘a good train journey’, NS increases the reliability of on time departure of trains, enough seating for passengers, and clean trains. To achieve this core focus, the first-line services on trains are important to guarantee the continuous and reliable deployment of trains, and the cleanness of trains.

Two major planning topics for NS are passenger transport and the maintenance of trains. The department *NS Operatie* is concerned with the planning of passenger trains. Roughly stated, they deploy the trains during the day over the main Dutch railroads. Planning of major and minor train maintenance is the responsibility of *NS Techniek* (former *NedTrain*). *NS Techniek* takes over the planning of the train as soon as the last passengers have left the train in the evening. Major maintenance tasks are performed on large maintenance stations and are planned far in advance. First-line services, the minor maintenance tasks, are executed at service stations. Service stations provide less services than the larger maintenance stations. These tasks are planned maximum one day in advance. In addition to the first-line services, service stations also facilitate the parking of trains during the night. This thesis focuses on the planning of the first-line services at service stations. In this thesis we abbreviate *service station* to *SB*, referring to the Dutch term *service bedrijf*.

The many SB locations are scattered throughout the Netherlands. All SBs have a different railway track layout and differ in the services they provide. The first-line services provided by an SB depend on the SB's facilities. For example, for the external cleaning of trains a special external cleaning facility is needed. These are not present at all SBs.

Some services, for example the inspection of a train, can be executed at all tracks. The necessity of the special facilities for certain first-line services complicates the service plan, because the trains need to arrive at the right track for the service. Compared to other vehicles, trains are very limited in their movements, because they drive on tracks. This means that the first-line service plan and the routing plan of the train units are highly interdependent.

The planning of first-line services is done by planners and superintendents in planning centers. Most of the planning is done manually, and based on logical thinking and previous experience. For now this works, but the number of passengers that use train transport is expected to continue to increase in the future. To cope with this increase of passengers, NS is expanding its fleet with three new train types (NS, 2015). This creates an even more complex planning situation, because more tasks need to be planned and executed with the same resource capacity.

To deal with these planning difficulties, NS currently develops two planning methods for their service station planning problem to support the planners. One system that NS develops to solve these problems is HIP (*Hybride Integrale Planmethode*), which is based on a heuristic. The other system that NS develops is OPG (*OPstelPlan Generator*), which approaches the problem in an exact way. Both systems use a different approach to find a feasible plan for the service station planning problem and are not used in practice yet. Furthermore, NS conducts ex

To create a feasible plan for an SB, a few steps need to be taken. Trains that are required for passenger transport are often a combination of different train units. First, these combinations are decomposed. Next, the trains are parked on the tracks of the SB. Then they can be driven to the tracks where the first-line service takes place, based on the first-line services plan. HIP considers in its plan creating the right train combinations, routing and parking the trains, and the first-line services. Since HIP uses a heuristic to create a plan, this plan does not have to be optimal. Therefore, the need arises for an exact method to find an optimal plan. NS realizes that developing planning methods is a major and complex task, thus multiple employees and scholars work together to improve and extend the planning methods. The exact planning approach that NS is developing currently, only deals with the routing of the trains. The first-line services plan is not considered in this method. Therefore, this thesis focuses on planning of first-line services at service stations to contribute to the overall development of the exact first-line services planning method at NS.

1.2 Research goal and research questions

Section 1.2.1 discusses the research goal and addresses the main research question. Section 1.2.2 provides the sub-questions and explains the approach for answering the sub-questions.

1.2.1 Main research question

The goal of this research is to contribute to the development of methods for the NS service station planning problem. We do this by planning the daily inspections, cleaning, and the minor maintenance tasks. Therefore, the research question of this study is:

How can first-line services at NS service stations be planned, using an exact approach?

As described before, the SBs vary a lot in track lay-out and in the services they provide. Since multiple studies focus on different subproblems of the service station planning problem, NS decided that all studies should be applied to the same SB. This is for integrating the subproblems and comparing solutions. Therefore, this study also focuses on this service station *Kleine Binckhorst*.

1.2.2 Sub-research questions

To answer the main research question, we answer a few subquestions. This section presents the sub-questions, describes the approach and states in which chapters we answer the sub-questions.

The first step is to get a clear understanding of the current planning process of the first-line services at NS. Therefore, the first sub-question is:

1. ***How are the first-line services currently planned at a service station?***

To map the current process of planning the first-line services at service stations, we shadow a superintendent on his shift, have multiple meetings with other planners and analyze company documents. Chapter 2 describes the planning process in the current situation.

After the current situation is clear, a literature study provides insight in framing the problem in the current literature. Then, we deduce approaches for solving such a problem from the literature. The related subquestion is:

2. ***What does the literature say about planning services and what exact approaches can be used for this?***

The literature review consists of an analysis of academic articles, PhD theses and books. See Chapter 3.

The third step is defining the problem clearly, and describing the model for planning the first-line services. Chapter 4 provides this, and describes the input data. The corresponding sub-question is therefore:

3. ***How can we model the first-line services planning problem?***

The fourth step is to analyze results of the model on service station *Kleine Binckhorst*. Chapter 5 answers this sub-question:

4. ***For which problem instances can we find a first-line services plan, using the model?***

Chapter 5 presents and analyzes the results of the model by solving different first-line services problem instances. Subsequently, Chapter 6 presents the conclusions and the discussion. Moreover, it provides the recommendations and suggestions for future research.

2 | Context Analysis

This chapter answers the first subquestion: *How are the first-line services currently planned at a service station?* This thesis takes SB *Kleine Binckhorst* as its case, as described in Section 1.2.1. Therefore, this chapter focusses on this SB.

Section 2.1 provides a short introduction on rolling stock, explains relevant railway sector terms and presents a short overview of the various NS maintenance stations. Section 2.2 describes the tasks that need to be planned at SB *Kleine Binckhorst* and Section 2.3 presents an overview of the layout of SB *Kleine Binckhorst*. Finally, Section 2.4 discusses the planning process.

2.1 Rolling stock and maintenance stations introduction

Figure 2.1 displays a picture of a train. A train is divided into subparts called train units.

Figure 2.1: train of train type VIRM

Rolling stock is used in the railway industry for all vehicles moving over railways. This term includes; trams, subways and trains. Trains can be divided into subparts, as just described, and these are called train units. Single train units or combinations of train units are deployed by NS for passenger transport. Single train units all have a unique train unit number. Train units can be further classified into parts (*bakken*), which is a measuring unit of length equaling 27.2 meter. This measuring unit is sometimes used to define the length of a track.

Furthermore, trains are of different types. Examples are: SLT (*Sprinter LightTrain*) and VIRM (*Verlengd InterRegio Materieel*). The VIRM can be viewed in Figure 2.1. Only train units of the same train type can be combined. The different train types all require different first-line services in different time intervals. For example one train type may need to be inspected every single day and another every two days, because of its technical features.

To perform the first-line services, there are different types of stations. The largest, and the most extensive station in providing maintenance services, is the refurbishment station (*onderhoudsbedrijf (OB))*. Here, train units come for large, planned maintenance. These maintenance tasks have a duration between two days to a week, depending on the train type and the required maintenance. Another reason for a train unit to enter ans OB is when unexpected failures occur, and repairs are needed. Less extensive versions of the OB are the technical centres (*technisch centrum (TC)*). These focus on providing quick repairs on frequent occuring train unit failures. The service station (*service bedrijf (SB)*) is the least extensive maintenance station. An SB provides the first-line services: the inspections, cleaning and small maintenance tasks (e.g. replacing a light bulb). SBs consist of maintenance tracks (*behandelsporen*) and stabling tracks (*opstelsporen*). The maintenance tracks have special features, for example a platform next to the track to facilitate internal cleaning tasks. The stabling tracks are for parking the trains during the night and the inspections. Finally, NS has also parks that consist of only stabling tracks. At these parks no services are provided; trains can only be parked for the night. In this thesis we focus on the SBs, because here the first-line services are executed.

2.2 First-line services

Train unit first-line services are documented in a system called Maximo. Maximo keeps track of the tasks that are performed on the individual train units, and determines the deadlines for the next (periodical) first-line services. When unexpected services are needed, train drivers and planners can manually insert a work-order in Maximo.

At SB *Kleine Binckhorst* train units are inspected, cleaned internally and externally, and small repairs are executed. These first-line services can include different types, for example, there are two types of inspections, inspection A and inspection B. Therefore, at SB *Kleine Binckhorst* there are nine first-line services that we consider. Every service has a time window in which the service needs to be completed. The time window for inspections A B is 24 hours, and the time window of the internal cleaning service is tree days. It varies per service what the consequences are if the deadline is not met. For example, if a train unit is not externally cleaned within the time window, the train unit can still be deployed for passenger transport. It gets a high priority to be cleaned at the next SB that the train unit enters. When an inspection A or B is not performed in time, the train unit is not allowed to leave the SB before the inspection is completed.

The next paragraphs describe the services that can be performed at *Kleine Binckhorst* more elaborately .

Inspection A and B

There are two inspections that can be executed at service station *kleine Binckhorst*; inspection A and inspection B. Inspection A is the larger inspection of the two and takes about an hour to execute, inspection B takes about 20 minutes. Inspection A needs to take place, approximately once every 12 days and inspection B once every two days. The exact durations and time intervals differ per train type. Also the content of the inspections vary per train type. We explain the inspections for the SLT (*Sprinter Light-Train*). We describe Inspection B first, because Inspection A is an extensive variant of inspection B. Inspection B consists of inspection preparation proceedings, brake testing proceedings, and external checks on the streamers (*stroomafnemers*) of the train unit (NT Operations, 2015a). Inspection A consists of the proceedings of inspection B, and a train driver cabin & passengers cabin check on interior, safety doors and other safety measures, and lightning. Finally, the bogies (*draaistellen*) are checked, as are the outside walls and the entry step (NT Operations, 2015b).

Internal cleaning

Internal cleaning can only be done when there is a platform next to the track, or when there are movable steps connected to the track. This is because of safety regulations for the cleaning personnel. SB *Kleine Binckhorst* is equipped with a cleaning platform with a track at both sides. Internal cleaning of train units is outsourced to cleaning company HAGO. NS is responsible for the planning of train units on the tracks next to the cleaning platform and HAGO cleans the train units. HAGO notifies the planner when the train unit is finished so this train unit can be driven to its next location on the SB, and a next train unit can arrive to be cleaned. HAGO starts during weekdays, around 21:30, and earlier during the weekend.

External cleaning

Train units need to be cleaned externally approximately every 12 days. To do this, there are washing stations, through which the train unit slowly drives. There are two cleaning treatment for the train units at the SB: a soap and an oxalic treatment. An oxalic treatment is more thorough than a soap treatment and also takes more time. The treatments are applied during two different cleaning programs. One is cleaning the train unit walls and the other cleans the front of the train unit (*kopwasbeurt compleet*).

Small repairs

Small repairs are repairs that do not need any special conditions to be executed. Examples include door repairs, or changing a bolt or light bulb. These repairs can be executed on the stabling tracks.

2.3 Service station lay-out

Service station *Kleine Binckhorst* is located near The Hague and is part of a bigger service station location. This location consists of SB *Kleine Binckhorst* and SB *Grote Binckhorst*. Because these two location are divided by main rail road tracks they are treated as separate service stations. Both SBs have a very different layout.

The layout of NS SBs can roughly be divided into two categories: *shuffleboard*-layout (*sjoelbak*-layout) and *carousel*-layout. Figure 2.2 shows the two layouts of which the top part is SB *Kleine Binckhorst* and the bottom part is SB *Grote Binckhorst*:

o At a *carousel*-layout, the tracks are *free* tracks, which means that trains may enter and depart the track at both sides. At a carousel layout a train units drive a round trip over the SB. A train may enter a track from one side, leaves on the other side of the track, and drives back to its starting point using a second track. These circle routes create the carousel-structure. However, train units may enter and depart from the same side of the track when this is required.

o A *shuffleboard*-layout includes tracks that train units only can enter and depart on one side. Trains enter and depart these tracks according to the ‘Last In, First Out’ (LIFO) principle. Trains are, *shuffled* onto a track and parked there until the train needs to move to a next track.

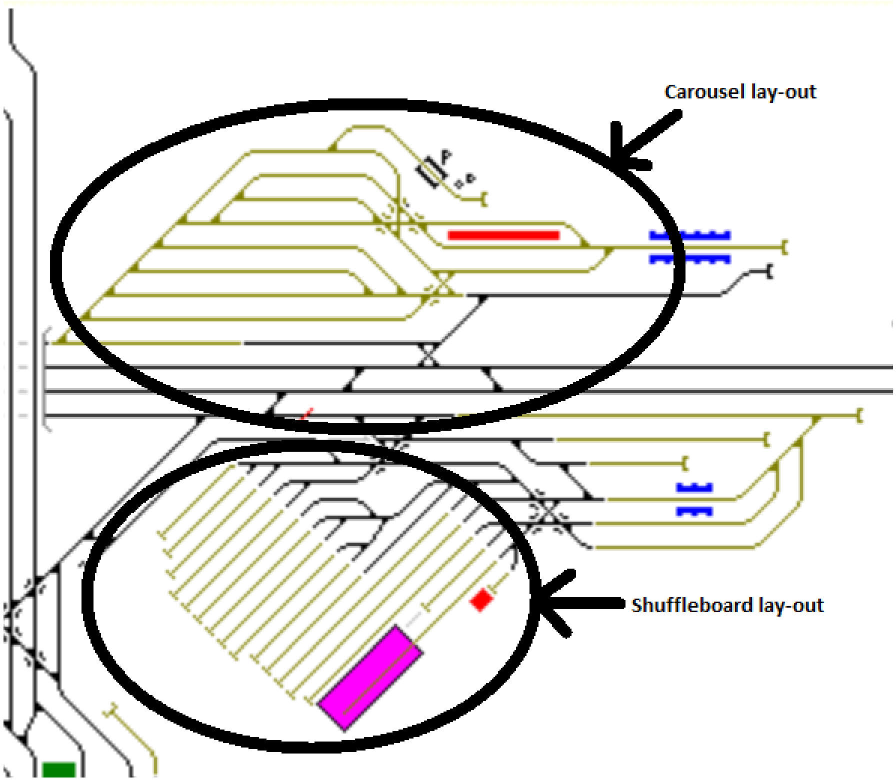

Figure 2.2: Shuffleboard- and carousel-layout

Figure 2.3 shows a map of SB *Kleine Binckhorst*. As can be seen in this figure, SB *Kleine Binckhorst* contains mostly free tracks. Only tracks 64 and 63 are tracks that can only be entered and departed from one side. Therefore, this layout corresponds with a carousel-layout.

Train units drive towards the SB *Kleine Binckhorst* on tracks 904, and 903b, because these are main tracks of the Dutch railroads. Main tracks are tracks between stations or towards SBs and OBs. From the main tracks the train units enter the SB using the entrance tracks: tracks 906a and 51a. Figure 2.4 highlights the main and entrance tracks.

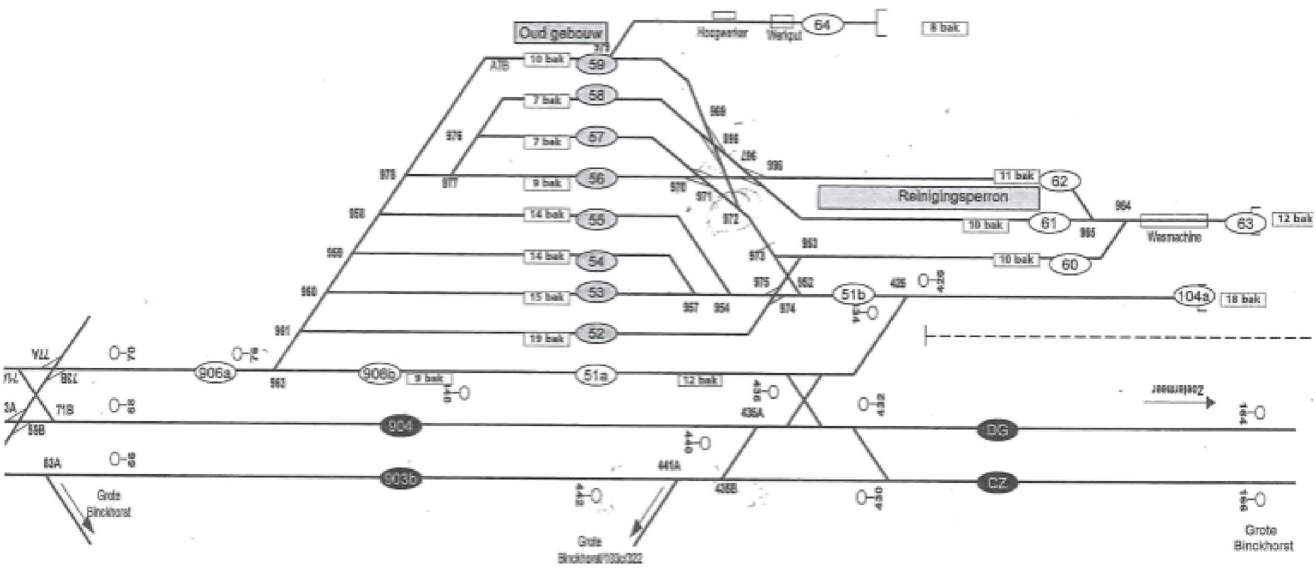

Figure 2.3: Layout of service station Kleine Binckhorst

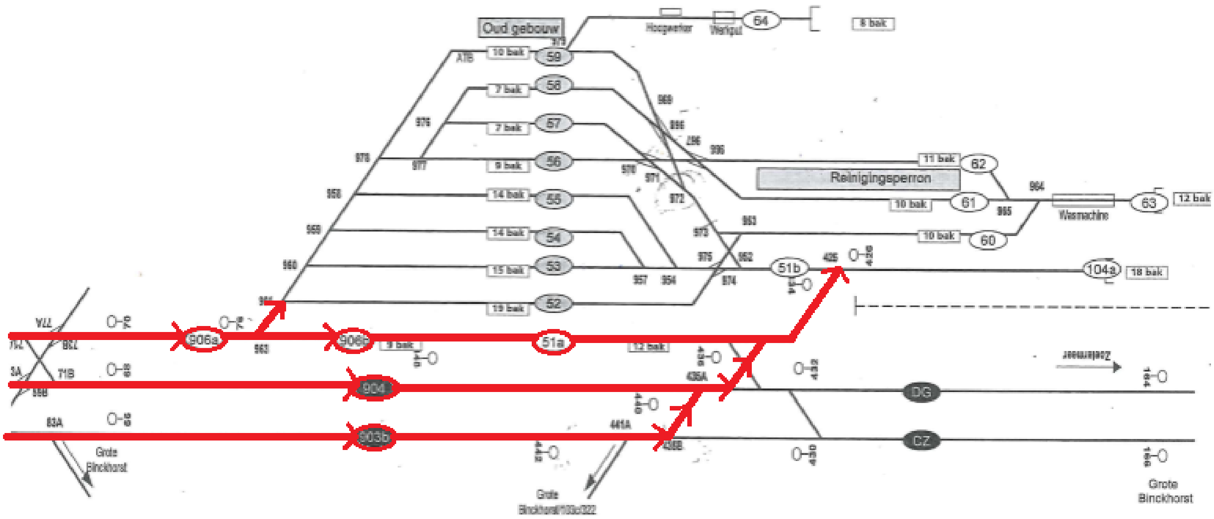

Figure 2.4: Entrance route for train units at service station Kleine Binckhorst

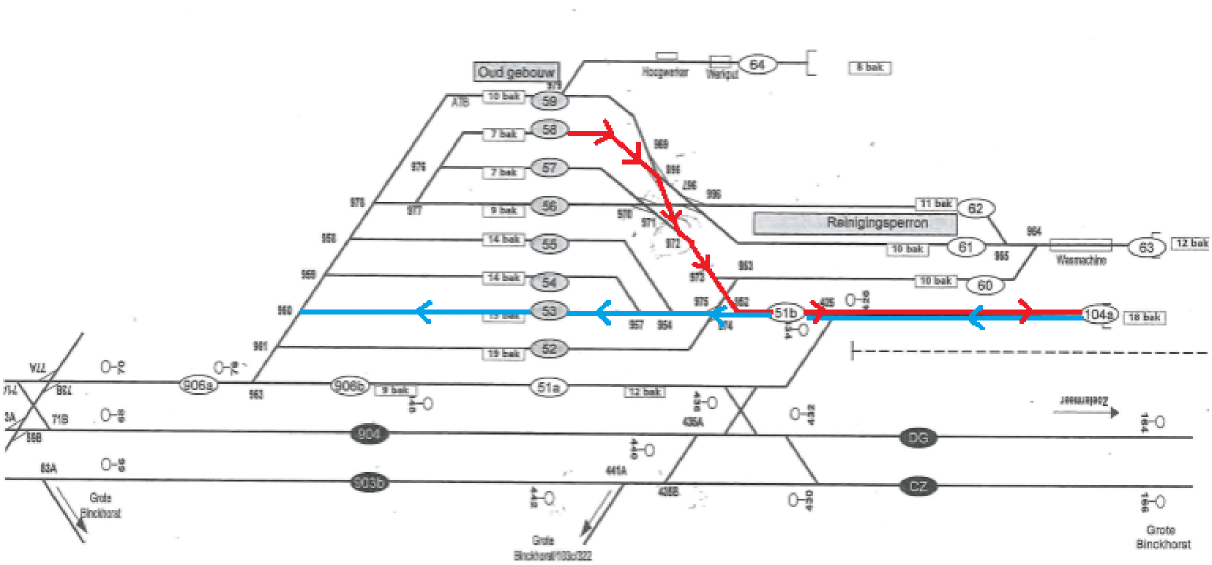

Figure 2.5: Route from track 58 to track 53

From tracks 51a and 906a, train units can reach all other SB tracks. However, not all tracks can be reached directly from all other tracks. The tracks in the SB layout that are connected to other tracks with an angle less than 90 degrees, cannot be reached directly by train units. In such a situation, a third track is needed. For example, if a train unit needs to drive from track 58 to track 53, it needs other tracks to reach track 53. One route is by using track 60, another route can be by using track 104a. The latter route is displayed in Figure 2.5 and explained in the next paragraph.

To come from track 58, the train unit crosses seven crossings before reaching track 51b. Because that is a very small track, the train unit likely needs to cross crossing 425 to enter track 104a. Here, the train unit is parked, because the direction of the train unit needs to change. The front of the train unit becomes the back, and vice versa. This process is called *kopmaken*. The train driver shuts off the train unit, walks to the other side of the train unit and starst the engine before crossing 425 is crossed again. Subsequently, the train unit passes track 51b and four crossings before the train can be parked on track 53. This drive takes a lot of time, because all crossings need to be adjusted to guide the train unit to the right track.

Table 2.1: Location of services at SB *Kleine Binckhorst*

External cleaning	The soap and oxalic cleaning treatments are provided on platform 63 in the cleaning station.
Internal cleaning	These tasks are executed at track 61 and track 62. Next to these tracks there is a cleaning platform that is needed for the cleaning personnel, because of safety regulations.
Repairs	Two places for special repairs are located at track 64. Track 64 contains a telehandler (*hoogwerker*) for repairs of the streamers and a work pit (*werkput*) for repairs on the bogie of the train unit. The work pit is little used. For bogie repairs the train units often go to a TC. Other kind of repairs are executed on all other tracks.
Inspection A and Inspection B	Inspections A and B may be executed on all tracks. The only exception is track 64, here, only B-inspection may be executed, but this rarely happens on this track.

2.4 Planning process

The planning of first-line services on the train units, the coupling and decoupling of the train units, and the assignment of train units to tracks are done by planners (*werkvoorbereiders*) in shifts. During the day, only a few train units are parked at the SB. The busiest moments are during the night, when most tracks are occupied by train units and the first-line services are performed on the train units. The parking of the train units and the planning of the first-line services for the evening and night are done during the *late* shift (14:00-22:00) of that day. During the day, the capacity is large enough to do without a plan. The planner monitors the incoming and outgoing train units real-time. The late shift planner plans for the end of the late shift, when it starts to get busy with incoming train units, and plans for the *night* shift (22:00-06:00). During the night shift, the planner is busy with monitoring whether everything goes according to plan. Figure 2.6 visualizes a simple case of the SB planning process.

There are two types of planners:

- The logistics-planner is responsible for assigning train units to tracks.
- The services-planner is responsible for creating the services plan, and assigning personnel to the services on train units.

In the planning process of parking train units and executing the required services, multiple parties and steps are involved and needed. First, when a train unit needs to enter the SB that evening, the logistics planner gets a notification. He checks if the train unit is defect and what the defect is. Based on the defects of the train unit and the availability of the tracks, the planner assigns a train unit to a track. He documents this decision in an overview of the track-train unit combinations. The services-planner can also access this overview and based on the unique train number of the train unit that has just been matched to a track, he searches in Maximo what the first-line services are that are required for this train unit.

The services-planner assigns a mechanic to the train unit-service combination, and updates this decision in another overview. When the train arrives at the SB, the train driver needs to stop before, so called, an S-sign, and needs permission of the train operator (treindienstleider) to enter the SB. The train operator has also access to the system with the train unit-track combinations overview. The train operator communicates to the train driver to which track he needs to drive, and whether it is safe to drive. Subsequently, when a first-line service is done, a sign that the service is completed is communicated both to the logistics- and the services-planner. The logistics-planner now decides to what track the train needs to be driven. Again, permission of the train operator is needed before moving the train unit. When all services are completed, the train unit is driven to a departure track from which it can leave the SB. This is in general the process that is executed for every incoming train unit.

The internal cleaning tracks are the planning bottleneck at SB *Kleine Binckhorst*. To drive to the departure tracks the train units need the external cleaning track (track 64 in Figure 2.3) to change direction. Because of this, only a few train units are cleaned externally every night. About 28 trains enter SB *Kleine Binckhorst* every day. Because not all first-line services can be planned, planners make decisions in which first-line-services to process and which to delay. The planning decisions are made based on the first come, first service principle, on the earliest due date principle, and based on the services with the highest priority. There is not a clear understanding of the capacity of the SB and currently all decisions are made on the planners experience. In addition to the general process that we just described, there are deviant scenarios that may occur. For example, there can be an extra, unexpected train needing to enter the SB, resulting in a more crowded SB.

Moreover, another train unit of the expected train type than expected may show up in front of the S-sign wanting to enter this SB. This train unit may have a totally different combination of services that are required. These changes in train units happen, because *NS Reizigers* (*NSR*) creates the timetable for the passenger transport. They decide on the train type and train trajectory combination. NS Techniek is subsequently responsible for delivering train units of that train type on the train trajectory. NSR tries to create such a timetable that train units finish the timetable near an SB. For the planning of the first-line services that need to be executed at SBs, it is important to know which train units will enter the SB. The capacity of two train units of the same train types are the same. So it does not matter for passenger transport which train unit is used, but it does matter for an SB. Two train units of the same train type might require very different first-line services to be executed during that night. This leads to changes in the first-line services plan.

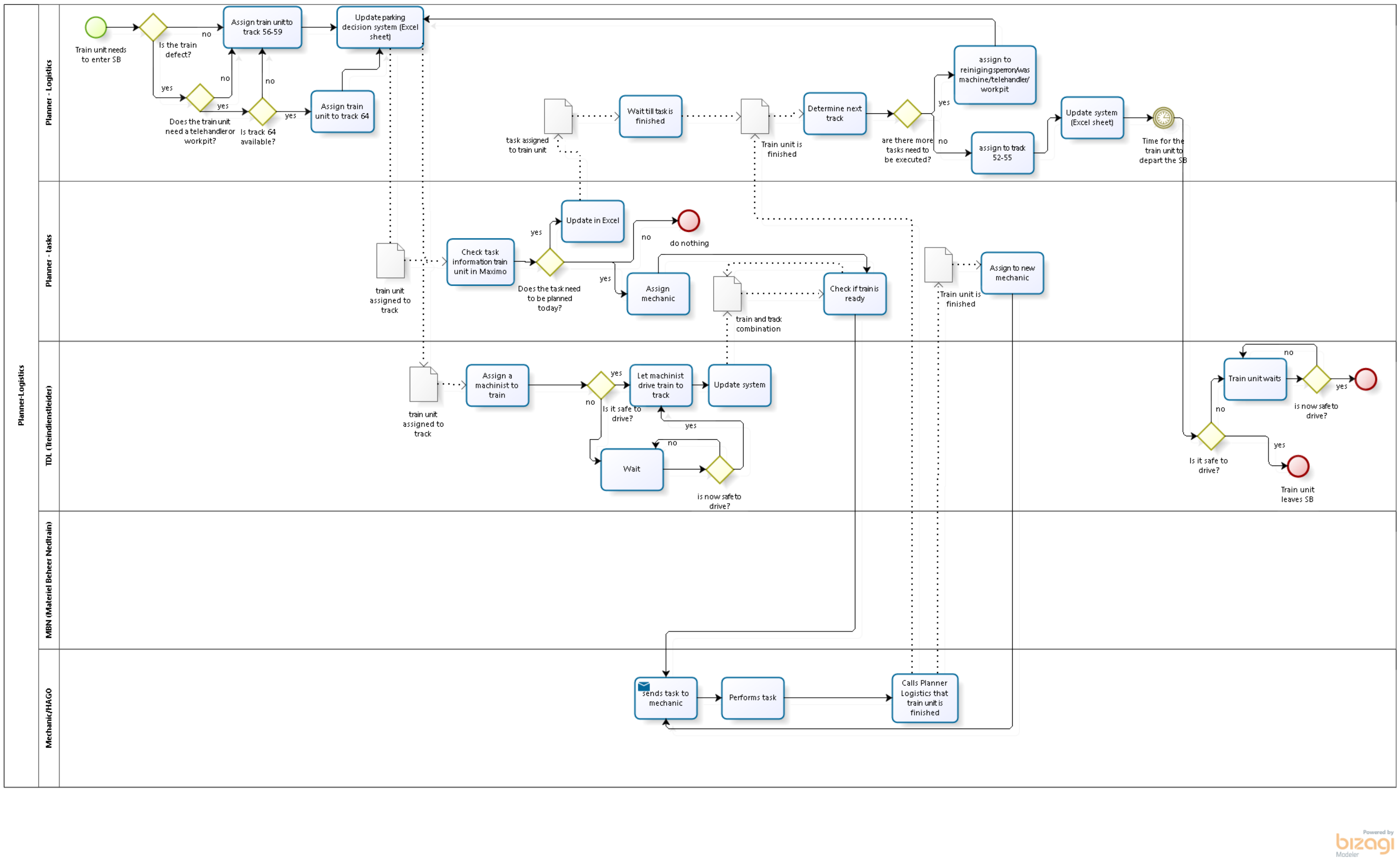

Figure 2.6: Planning process of train units at service station *Kleine Binckhorst*

3 | Literature review

This chapter frames the first-line services planning problem in the literature and discusses approaches for solving such a problem. Hereby, answering the second subquestion as described in Section 1.2.1: *What does the literature say about planning services and what exact approaches can be used for this?*

This chapter is structured as follows: First, Section 3.1 provides an introduction on planning rolling stock. Section 3.2, Section 3.3 and Section 3.4 describe the three pillars for creating an NS service station plan: the routing of train units, crew scheduling, the planning the first-line services on non-human resources. Section 3.2 describes the first pillar, the routing of trains, also called the Train Unit Shunting Problem.

Section 3.3 describes the difficulties related to the second pillar, crew scheduling. The third pillar is the planning of first-line services on the available, non-human resources. Section 3.4, describes approaches for planning resource capacity. Subsequently, Section 3.5 describes solution approaches for planning problems. Finally, Section 3.6 provides the conclusion of this chapter.

3.1 Rolling stock planning

Rolling stock planning problems are extensively researched. Studies on planning are conducted in the fields of creating passenger transport timetables (Yang et al., 2009), routing of rolling stock (Cadarso and Marín, 2010; Fioole et al., 2006; Wagenaar et al., 2017), and the railway maintenance of rolling stock (Albrecht et al., 2009; Peng and Ouyang, 2014). For the planning of rolling stock maintenance Sriskandarajah et al. (1998) present a genetic algorithm. This algorithm optimizes the maintenance overhaul, Cheng and Tsao (2010) extend this algorithm by also taking into account the spare parts that are needed for maintenance tasks. Penicka et al. (2003) create a formal model of the train maintenance routing problem. Corman et al. (2017) study and use preventive maintenance to determine the optimal maintenance policy for a light rail rolling stock system in terms of reliability, maintenance costs, and availability. So, many aspects in the field of rolling stock planning problems are research. However, not many studies are conducted on planning the first-line services of rolling stock. Giacco et al. (2014) present an optimization framework for rolling stock rostering and maintenance scheduling. These studies on solving rolling stock maintenance take maintenance tasks in account that can be described as planned maintenance. The first-line services plan cannot be made far in advance and has little flexibility in which services to execute and which not. Therefore, the plan needs to include as many required services as possible.

The first-line services planning problem, is on the level of offline and online operational planning. Plans on offline operational level are plans that are made short in advance and online operational plans are made based on reactive decision making (Hans et al., 2012). As described in Section 2.4, the NS first-line services plans are made in advance and are adjusted real time, when needed.

Service station planning problem is a very complex problem in terms of size. Therefore, this problem is by scholars often divided into subproblems. These subproblems are subsequently solved independently. Table 3.1 presents the different subproblems that are defined in the planning of daily maintenance on service stations.

Table 3.1: Subproblems of the service station planning problem

Subproblem	Definition
Train Unit Shunting Problem (TUSP)	The TUSP is concerned with the assignment of tracks to trains. The TUSP can be further decomposed in the matching and parking problem (Freling et al., 2005).
- matching problem	The matching problem is about deciding when and where to decouple and couple which train units, to create the train unit combinations that are required for the departing trains (Freling et al., 2005).
- parking problem	The parking problem is about where to park the trains at the SB (Freling et al., 2005).
Crew scheduling problem	The crew scheduling problem is about creating rosters for crews that include the assignment of crews to train units while satisfying regulations and union work rules (Bojovic and Milenkovic, 2010)
Resource planning problem	The resource planning problem is about efficient and effective utilization of resources, in such a way that a realistic plan can be formulated, and the bottlenecks and the conflicts can be identified (association for Project Management, 2017).

The subproblems listed in Table 3.1 are described more elaborately in Sections 3.2, 3.3 and 3.4. This is important in understanding the context in which this study is placed. The first-line services plan is one of the subproblems and is highly interrelated with the other subproblems, as will become clear in the following three sections.

3.2 Train Unit Shunting Problem

A service station can also be called *'shunting yard'* and provides the first-line services. It contains tracks over which the trains drive. The movement of a train from one track to another on a service station is called shunting. The shunting problem tries to minimize the number of movements in a shunting yard. One of the first works on this shunting problem is by Blasum et al. (1999). They focused on the parking and the dispatching of trams with minimum shunting movements. Gallo and Miele (2001) apply the shunting problem to a bus depot, and try to minimize the movements of buses in a small and crowded depot. The buses may not arrive in the planned order and decisions have to be made with incomplete information. Varying the length of buses that need to be dispatched is the solution that Winter and Zimmermann (2000) propose to make the solution of Gallo and Miele (2001) more realistic.

The train unit shunting problem (TUSP) is a special kind of shunting problem. The difficulty with vehicles that drive on tracks is that they are bounded by the tracks in the possible directions that the vehicles can move. It means that rolling stock can only drive in one direction in a two-dimensional plane. Cars or trucks, opposed to rolling stock, are much more flexible in the directions that they can drive. They can drive in any direction in the two dimensional plane. Another difficulty with rolling stock is that trains cannot bypass other trains. For example, to change the order in which they are parked, also because they drive on tracks. Finally, trains have large turn angles. This may prohibit trains to directly reach certain track from its current position on the SB. Therefore, to reach this track, a third track is needed, as described in the example provided in Section 2.3. All this factors complicate the routing of trains extremely.

Freling et al. (2005) introduce the distinction of the TUSP subproblems as explained in Section 3.1. They propose a MIP model for the matching problem and the parking problem based on column generation. Cornelsen and Di Stefano (2007) solve the parking problem by using a conflict graph and Di Stefano and Koči (2004) arrange train units that are needed to be dispatched in the morning in such a way that no shunting movements are needed. For this, they propose different heuristics to solve different problems in terms of allowed arrival and departure directions on tracks. For example, train units may only arrive on one side of the track, and depart on the other side of the track, or train units are allowed to arrive and depart from both sides of the track. Wolfhagen (2017) conducted a TUSP study on a shunt yard of NS. She based her study on the approach of Freling et al. (2005), and extends the TUSP by allowing each train unit to be reallocated once during its stay at the SB. She defines this extension as the TUSP-R.

3.2.1 Matching problem

Trains arrive at an NS SB as a single train unit or in a combination of two or more train units. These train units are of the same train type. The required compositions of departing trains the next day, may be different than the compositions of the arriving trains. An example: two trains arrive, both consisting of two VIRM train units. For the departing trains, one train consisting of three VIRM train units and one train consisting of a single VIRM train unit are required. To change from the arriving train unit compositions to the required departing compositions, one of the two VIRM train unit combinations needs to decouple into two single train units. The other VIRM train unit combination needs to couple to one of the two single train unit.

Transforming the arriving train unit combinations into the required departing train unit combinations is called the matching problem. To facilitate this transformation, decisions need to be made on which train unit combinations to decouple, which to couple, and when and where to execute these actions. These decouple and couple movements may take place on every track and each moment during the stay of the train unit on an SB. Fioole et al. (2006) solve this problem while considering multiple objectives as operational costs, service quality and reliability of the railway system. Also Peeters and Kroon (2008) solve this problem by developing a branch-and-price algorithm that is performs well on the objectives 'service to the passengers', 'robustness', and 'cost of the circulation'.

In making these decisions, the first-line services plan needs to be considered. It might be more efficient to process certain operations on the whole arriving train, and to split then, or the other way around. Also, in matching train units, the track length needs to be taken into account, to make sure the combination of train units fits the track. Since the tracks are not of equal lengths, this leads to the parking problem, described in Section 3.2.2.

3.2.2 Parking problem

The parking problem, also called the '*track assignment problem*', is the problem of assigning train units to tracks. The parking problem takes into account both the stabling and the maintenance tracks. The objective of this problem is to create train unit-track combinations for all train units on the SB, in such a way that the total shunting movements are minimized. This plan needs to meet the track length restrictions. Not all tracks are equal in length, and not all tracks are equipped with overhead wires that provide the train units that need it, with electricity. Furthermore, the train units need to be planned such that the first-line services plan can be executed. The order of train units on tracks is particularly important to fulfill this condition. Here, a compromise between shunting movements and on time first-line services need to be made.

The matching and parking problems have been integrated by different scholars. Lentink et al. (2006) created a 2-opt heuristic to improve the route that was found by sequentially solving the subproblems. Haijema et al. (2006) present a dynamic programming based heuristic for integrating the matching and parking problem. Kroon et al. (2008) use an NS case to solve the matching and parking problem, hereby focusing on reducing the computational time for generating an acceptable solution.

For every shunting movement that needs to be made, train drivers need to be available which complicates the feasibility even further. In Section 3.3 we elaborate on crew scheduling.

3.3 Crew scheduling

Crew scheduling is about assigning the human resources to the first-line services. For both inspection A and inspection B, mechanics are needed to execute these services. Furthermore, a cleaning crew is needed for internal cleaning. In addition to the employees that are needed for the first-line services, also train drivers are needed for driving the trains to, on, and from the SB. Train drivers have different authorizations, so different employees need to be scheduled for different driving tasks.

The employees work in shifts and have different skills, availabilities, and authorizations. All these factors need to be taken into account when planning the human resources. Furthermore, the train units change position in the SB during their stay. Therefore, the planning of the personnel is complicated by traveling time of the personnel from one service to another.

Multiple papers are written about this problem. Dutot et al. (2006) define the crew scheduling problem as the technician and task scheduling problem (TTSP) and describe the challenges of this problem elaborately. Fırat and Hurkens (2012) provide a MIP formulation for the TTSP and Cordeau et al. (2010) approach the problem with an adaptive large neighbourhood search approach. Kovacs et al. (2012) extend the solution approach by Cordeau et al. (2010) by adding release and due dates and geographical locations to the problem, calling it the service technician routing and scheduling problem (STRSP). Finally, (den Ouden, 2018) generates robust crew schedules for maintenance staff using a greedy heuristic to find an initial solution and improves this solution by a local search heuristic, while focusing on fairness, flexibility and walking distances. This study by den Ouden (2018) is also applied to SB *Kleine Binckhorst*.

3.4 Resource planning

Another important part of the service station planning problem is the scheduling of the resources. Every service that needs to be performed requires resources, e.g., a cleaning machine, personnel, or a platform. These resources need to be used as efficiently as possible to minimize costs and to maximize productivity. In this section, we focus on the non-human resources.

All services that need to be performed during one night on a train unit can be viewed as a project. The different train units that enter the SB on the same day are all projects that consists of activities. Therefore, the planning of these activities can be defined as a Project Scheduling Problem (PSP). Al-Fawzan and Haouari (2005) define project scheduling as dealing with "the allocation of scarce resources to a set of interrelated activities that are usually directed toward some major output and require a significant period of time to perform". PSPs contain activities that can be planned simultaneously or sequentially, depending on the available resources and the completion of predecessor activities. PSPs are widely studied problems.

Another way of regarding the planning problem is as a job shop scheduling problem. These are a special case of machine scheduling problems, and are found in the production environment. These problems are already a long time a popular area of research in the operations research literature and production literature (Schutten, 1996) and still are Wang (2005). The operations research studies on job shop scheduling mostly focus on achieving a high level of algorithmic design and analysis, and studies in the production area emphases the problem formulation and providing practical solutions. Job shop problems can be a way to model rolling stock problems according to Samà et al. (2016). They used the job shop problem for finding a lower bound for a train routing problem. The flexible flow shop problem is a specific variant of the job shop problem, which is used by (van Dommelen, 2015) to model the cleaning facilities on an SB. The next section elaborates on this.

3.4.1 Flexible Flow Shop Problem

Before describing the classical job shop problem and its extensions and variants, we need to point out that in the terminology used in job shop problems and PSPs, different terms are used for the same concept. In job shop problems a PSP *project* is defined as a *job*. In job shop problem literature an *operation* equals the PSP *activity*. In this thesis we use the terms *job* and *operation*. So, one job consists of one or more operations.

In the classical job shop problem jobs have operations that need to be processed in a fixed order on different machines. Every operation requires one specific machine. The jobs and machines all become available for processing at the same time. The objective of the classical job shop problem is to minimize makespan (Schutten, 1996). Because in reality many more factors need to be taken into account, extensions are developed.

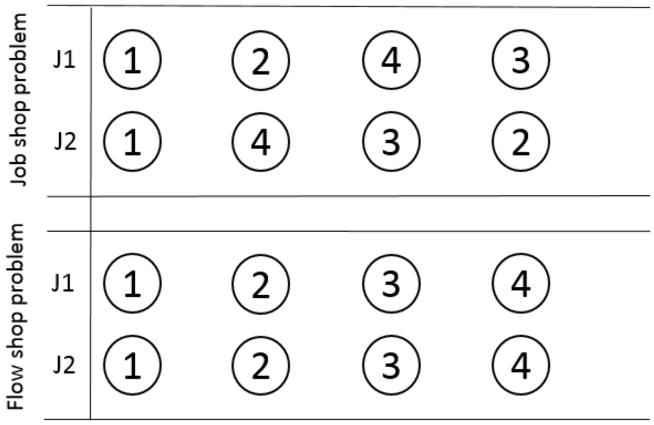

Figure 3.1: Order of operations

One special case of a job shop problem is the flow shop problem. The difference between these problems is in the order in which operations need to be processed. In a job shop problem the order of processing of operations may differ per job, but within a job the order of processing operations is set in advance. In a flow shop problem the order of processing the operations is the same for all jobs. These differences are visualized in Figure 3.1. In the example in Figure 3.1, both the job shop and the flow shop problem have two jobs consisting of four operations. In the job shop scenario, job 1 has to process the operations in the order 1-2-4-3 and job 2 in order 1-4-3-2. Of both jobs all operations need to be processed before the jobs are finished, but the order of processing is different. In the flow shop problem scenario, the order of operations of both job 1 and job 2 are equal.

Within the flow shop problem different variants can be distinguished. One variant is the *flexible flow shop problem*. Flexible flow shops are also called *compound*, *hybrid* or *multiprocessor flow shops*. In a flow shop problem, all jobs have operations in multiple, consecutive stages, that all need to be processed in a fixed order. Just like a job shop problem, only one machine is available per stage. In a flow shop problem every job only has one operation per stage. The flexible flow shop problem comprises a more elaborated problem. A job goes through multiple stages and there are one or more identical machines available at every stage. These machines are set in parallel. A given job has only one operation per stage that needs to be processed on one of the parallel machines in that stage. Every machine processes at most one operation at the time and the processing time is assumed to be deterministic and integer. All machines are ready from time zero onwards (Haouari et al., 2006).

The flexible flow shop problem is still very general compared to practice. Therefore, also for the flexible flow shop problems extensions are developed. In flexible flow shop problems it is possible to skip a stage and not perform an operation at that stage, because not all jobs always need to be processed at all stages (Ruiz et al., 2008). Furthermore, in the flexible flow shop problem there is unlimited buffer capacity between the stages. Since in reality there are situations where no buffer capacity is present, or the jobs needs to be processed directly on the next machine, the no-wait flow shop is developed (Ruiz and Vázquez-Rodríguez, 2010). In the no-wait flow shop, the jobs need to be continuously processed until all jobs are finished, without interruptions. This means that the total processing time of a job is equal to the sum of the processing times of all the stages that that job has operations at (Jafarzadeh et al., 2017). For example in the food industry, food products need to be bottled or canned right after cooking so that the products are hot and fresh (Jafarzadeh et al., 2017). Also release and due dates can be taken into account as is the recirculation of jobs, meaning that they revisit a stage (Ahonen and de Alvarenga, 2017).

3.5 Solution procedures

Project scheduling problems and job shop problems can be solved by either an exact method or using a heuristic. RCPSPs are NP hard problems. Therefore exact solving is possible according to Brčić et al. (2012), but only for small to medium instances. They propose branch and bound as the most commonly used, and appropriate method for solving RCPSPs in an exact way. Still heuristics are needed to obtain an upper bound; the lower bound can be determined using mathematical programming (Bellenguez and Néron, 2004). Mixed Integer Programming is an approach to model the RCPSP and the job shop problem and can subsequently be solved using the branch and bound approach, or branch-and-price or branch-and-cut approach for example. Finally Constraint Programming and Satisfiability Testing are methods for solving the RCPSP. The combination of the two methods have proven to be very effective (Schutt et al., 2011).

We just described that RCPSPs can be solved exactly, however, heuristic methods for solving the RCPSP dominate the RCPSP research field (Brčić et al., 2012). Heuristic methods do not find the optimal solution but can approximate the solution quickly. The simplest methods use constructive heuristics that contain a priority list, for example serial or parallel schemes, but these heuristics find bad solutions (Trautmann and Baumann, 2009). More clever heuristics as the heuristic developed by Kolisch and Drexl (1996) find very good solutions, and dominate other heuristics that are developed for the RCPSP. Their heuristic uses a combination of priority rules and random search techniques. Newest approaches to solve PSPs incorporate machine learning (Jędrzejowicz and Ratajczak-Ropel, 2014).

So, both exact approaches and using a heuristic can solve a PSP problem. Exact approach can provide some information about the gap between the optimal solution and the current found solution. However the heuristics approaches dominate the RCPSP research because they can found good solutions very quickly.

3.6 Literature study conclusions

Answering the subquestion of this chapter, we conclude from the literature study that rolling stock is a theme that is highly researched. The emphasis of most studies is on routing problems and not much research is conducted on the first-line services planning problem. Because all problem are interrelated this is an important aspect of the service station planning problem. Focusing on the service station planning problem of rolling stock, the literature shows that this problem is often decomposed in different subproblems. These subproblems are solved independently, because they are all so complex.

Resource planning problems can be defined as a project scheduling problem or as a job shop problem. Both have multiple extensions to model the problem as close to the real situation as possible. Both can be solved by either an exact method or using a heuristic. The size of the problem is often the problem when solving an exact problem, however, when using a heuristic you do not know if the solution is optimal.

4 | Problem description

Chapter 2 explains how the first-line services plan is created currently, and Chapter 3 describes studies on the subproblems of the service station planning problem, and which solution approaches have been developed in previous research. These two chapters answer the first two subquestions. This chapter answers the third subquestion: *How can we model the first-line services planning problem?* It describes how we model the first-line services planning problem applied to SB *Kleine Binckhorst*, and how we can describe this problem mathematically.

This chapter is structured as follows: Section 4.1 describes the goal and scope of the thesis in more detail. Section 4.2 describes the modeling approach, and Section 4.3 states the input data. Subsequently, Section 4.4 provides an overview of the problem statement and describes the assumptions and constraints that are made to create the model. Section 4.5 presents the indexes, parameters and variables of the model, and Section 4.6 presents the constraints of the model. We describe Section 4.6 by using a running example. Finally, Section 4.7 provides the conclusions on this chapter and some remarks on the running example.

4.1 Goal and scope of the research

This section elaborates on the goal and scope of the model that develops a first-line service plan. Section 4.1.1 clarifies how the model helps to reach this goal and Section 4.1.2 elaborates on what we include and exclude in the model.

4.1.1 Goal

The overall goal of this research is to plan the first-line services for an NS service station using an exact approach. Currently, all tasks are planned manually, see Section 2.4. Often, not all tasks can be executed in one night. In the future the SBs will get even busier, because NS expands its fleet by over 250 train units. By modeling the situation of SB *Kleine Binckhorst* using an exact approach, we find (optimal) plans for the first-line services for different instances on this SB, within the assumptions and constraints that Section 4.4 describes. An exact approach can provide optimal plans, which HIP, as Section 1.1 describes, cannot. Optimal plans provide the best way of processing train units with their required first-line services, which is what NS want to achieve. In addition to this, the optimal solutions can provide insights in the quality of the solutions of HIP

4.1.2 Scope

From the different subproblems of the service station planning problem, we focus on the resource planning problem and leave all other subproblems out of scope. We assume all trains that enter the SB are single train units, because of this, we do not include the matching problem. However, the routing of train units on an SB is such an important and integrated subproblem that we do not want to disregard it totally. Wolfhagen (2017) developed a model to solve the TUSP-R for NS, but integrating this model with the first-line services model that is developed, is out of scope of this research project. This is because of the complexity of developing just the first-line services planning problem, and the time limit related to this research project. We choose to include the travel times between the different machines on which the operations are processed. This way, time for traveling between the machines is included in the first-line services plan. This is a step towards implementing the first-line service planning model with the routing planning model. In practice, the route of the train unit is reserved for the whole travel time for that train unit, for safety reasons. In our model, we leave this out of scope.

Also, the parking problem is a subproblem of the SB planning problem. We take one aspect of this subproblem into account, meaning, we assign tracks to train units in our plan. However, we do not consider the sequence of train units on tracks, because we simplified the model by restricting the tracks to contain at most one train unit at all times. For the crew scheduling subproblem, only one cleaning crew is available for internally cleaning the train units. We also include this in our model.

4.2 Modeling approach

Job shop problems and flexible flow shop problems are originally found in production environments. When analyzing the setup of an SB, many similarities between the SB situation and the flexible flow shop are found. In a flexible flow shop one has, as Section 3.4.1 describes, multiple jobs. These jobs all contain a set of operations that need to be planned on the machines that can process these operations. When translating the actual situation at SB *Kleine Binckhorst* into a flexible flow shop model, the arriving train units become the jobs. These jobs contain a list with first-line services, the operations, that are required for that train unit. The tracks of SB *Kleine Binckhorst* are the machines on which the operations need to be processed. There are one or more machines that can process the same operation. These machines are grouped, and can be uniform or have different features within a machine group. In our situation all machines within a group are uniform. Due to modeling the first-line services planning problem as a flexible flow problem, the sequence in which the jobs visit the groups of machines is equal for all jobs.

We refer to the groups of machines as stages. Furthermore, we aggregate the first-line services that can be processed on the same machine into one operation. This way the stages relate one-on-one with the operations. At SB *Kleine Binckhorst*, there are five operations, and thus five stages. Since in the service station planning problem, the shunting movements are minimized, it is very unlikely that first-line services that can be processed on the same machine, will be processed on different machines. Moreover, merging the first-line services into fewer operations makes the model easier to solve. Note that, it means that all bundled first-line services in one operation, need to be processed continuously on the machine, without interruptions of other jobs.

At SB *Kleine Binckhorst*, nine first-line services can be processed. Of these nine first-line services, a train unit requires at most seven, because only one of the two the inspections, and one of the two external cleaning services are required at a time. We aggregate these nine first-line services into five operations. Table 4.1 displays the operations and of which first-line services they consist. The table also shows the number of machines on which that operation can be processed and what the machine numbers are of the machines that can process that operation. These machine numbers help in understanding the next Figure 4.1, which is explained in the next paragraph.

In our model, not all jobs require seven out of nine first-line services. Jobs may require only 3 or 4 first-line services. Two first-line services is the minimum number of first-line services that a train unit requires, because the arrival and departure operations always take place. For the first-line services that are not required for the job, a processing time of zero is assigned to that first-line service. The processing time of an operation is the sum of the processing times of the first-line services that this operation contains. If the operation has a total processing time of zero, the train unit still visits this stage with machines, because of the carousel layout. This is because the stages are located in a sequence at the SB. Most stages cannot be skipped in order to reach the stage after that. Therefore, for simplicity we assume a job needs to visit all stages, even though the job is not processed at that stage.

The fixed sequence in which jobs visit the stages, and thus the sequence of processing the operations of a job, is a result of modeling the problem as a flexible flow shop problem. Having this fixed sequence limits the number of different plans that are considered by the model. This choice is a realistic limitation for SB *Kleine Binckhorst*, because SB *Kleine Binckhorst* has a carousel layout. In practice, in a carousel layout, the operations of all jobs are also processed in an (almost) fixed order. We could have disregarded this fixed sequence of processing operations, because now we do not consider all possible plans. However, these plans are likely not to be optimal for a carousel layout SB, especially when travel times and reservations of tracks may be added in the future. This is because the service station planning problem's objective is to minimize the number of shunting movements. The limitation that train units cannot overtake each other on a track also needs to be kept in mind. This makes traveling over the carousel layout in the opposite direction very difficult. Finally, by using the fixed route over the SB, the planners recognize the planning approach of the model, which helps in accepting the model, making it easier to implement the model. Based on these reasons, we choose to include the fixed order of processing operations.

Table 4.1 displays an overview of which first-line services that are aggregated into which operation and which machines process these operations. Figure 4.1 graphically visualizes the stages, the groups of machines, and the assigned sequence in which the jobs visit the stages. In the horizontal direction the different stages are displayed. Stage 1 and stage 5 are merged in the figure, because they consist of the same machines. Stage 1 and 5 contain the stabling tracks on which the train units arrive, the inspections take

Table 4.1: Overview of which the first-line services form the operations and which machines process which operation

First-line service	Operation number	Number of machines that can process this operation	Machine numbers
Arrival Inspection B Inspection A	1	8	1,2,3,4,5,6,7,8
Telehandler service Work pit service	2	1	9
Internal cleaning	3	2	10,11
External cleaning normal External cleaning oxalic	4	1	12
Departure	5	8	1,2,3,4,5,6,7,8

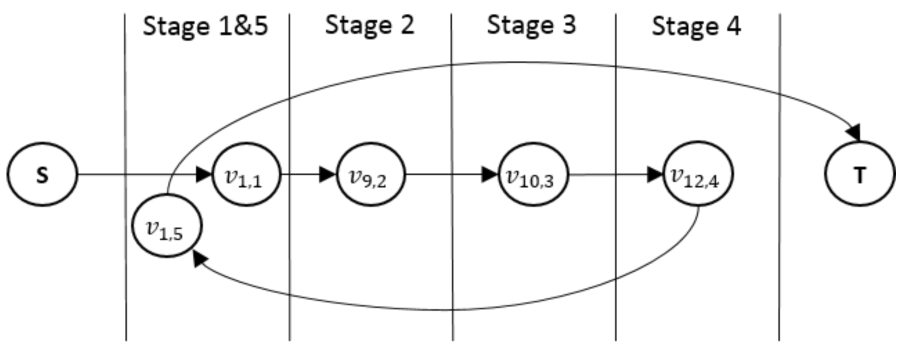

Figure 4.1: Graph representing SB *Kleine Binckhorst* as a flexible flow shop instance

place (operation 1), and the train units depart (operation 5), hence the merging of the two stages. The nodes in the graph in Figure 4.1 represent the events of this instance. The *s* and *t* in the first and last node represent the start and end event.

The nodes in the stages represent one machine per stage. We do this to make the figure easier to understand, even though in practice there may be more machines per stage, as Table 4.1 shows. The nodes are described as $v_{k,o}$ nodes of which the k stands for the machine number and the o for the operation number that is processed. The machines all have unique numbers. In the model the k represents the first machine number in that stage. So, for stage 1 that is machine 1, and for stage 2, it is machine 9. Stage 1 consists of eight machines, so machine 9 is the next machine number. Figure 4.2 displays the location of the different stages on the map of SB *Kleine Binckhorst*. In the right corner of the figure the legend is located. The legend relates the operation with the color of the stage. Since the operations and stages correlate one-on-one, the operation number equals the stage number. The operation in the figure is the aggregated form of the first-line services. From now on, we make a distinction between *operation* and *first-line service*. When we mean the first-line services, we will use *first-line services*. When we use the aggregated form of first-line services, we use *operation*.

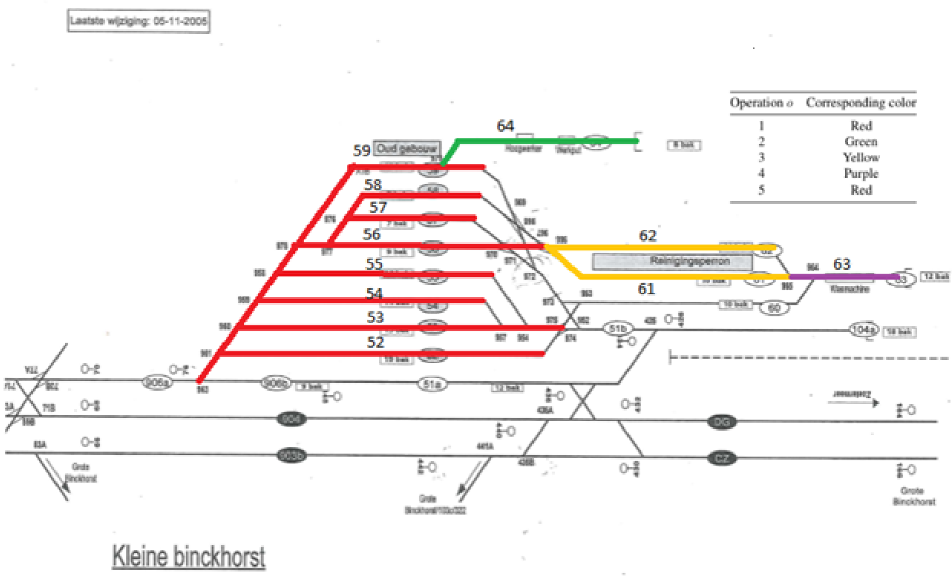

Figure 4.2: Tracks classified in stages

The following paragraphs describe the number of machines in every stage and the operation that is processed in that stage, according to the situation of SB *Kleine Binckhorst*.

Stage 1 contains eight machines, representing the eight stabling tracks of SB *Kleine Binckhorst*. Operation 1 is processed in this stage, meaning, the train units arrive on these tracks, and inspection A and inspection B are processed on the machines in this stage.

Stage 2 contains only one machine and processes operation 2 which consists of the first-line services: telehandler service and work pit service.

Stage 3 contains two machines, representing the tracks on both sides of the platform on which train units can be cleaned internally (operation 3). There is only one cleaning crew available at the time. Therefore, only one train unit on one of the two tracks can be cleaned at a time. Another train unit can be parked on the track but it has to wait on being cleaned, until the train unit at the other track is finished. In our model a train unit can stay on the track after being processed, but cannot arrive early. Section 4.6.2 describes how we model this.

Stage 4 contains the machine that externally clean the train units. Operation 4 is processed in this stage. Operation 4 consists of either the external cleaning normal, or the external cleaning oxalic service. This is because at most one of the two external cleaning options is required for a train unit per visit at an SB.

Stage 5 contains the same machines as stage 1, because the train units need to depart from the same tracks as the train units arrive and are inspected on. So, the capacity of the tracks need to be divided between operation 1 and operation 5.

TO answer the subquestion that we answer in this chapter, we formulate our problem as a Integer Linear Programming (ILP) problem. Integer linear problems are of the form:

$$\begin{aligned} & maximize\ c^T x \\ \textit{subject to:} & \\ & Ax \leq b \\ & x \geq 0 \end{aligned} \tag{4.1}$$

We choose this formulation, because the values of the variables are integer. we cannot assign fractions of a train to a track. Furthermore, all constraints need to be, and are, linear. This reduces the solving time of the model compared to a non-linear model.

To summarize, we group the machines on SB *Kleine Binckhorst* such that the machines that can process the same operations belong to the same group. Furthermore, first-line services that can be processed on the same machine, are bundled into one operation. This operation is continuously processed on the assigned machine. Due to this approach, the operations and the stages correlate one to one. We model the ILP in AIMMS Optimization Modeling. The model is solved in AIMMS with a CPLEX solver by using the branch-and-bound approach. While solving the model, nodes are created that represent possible solutions that are compared with future solutions to find the optimal one. The CPLEX solver calculates an LP bound, and as long as the optimal solution is not found yet, an integrality gap is calculated.

4.3 Data

The input data for the model are the following:

1 A table that contains which operation can be processed on which set of machines at SB *Kleine Binckhorst*.

2 A table that contains the processing time of every operation for each job, based on the required first-line services and the train type of the job. The train type is important, because the processing times of the first-line services are different for all train types.

3 As described in Section 4.1.2, we include the travel time between all machines. The input for the model is a table that contains the travel times between all machines. Since we use a fixed sequence in which the machines visit the stages, not all travel times are needed. However, if the model is to be extended and these are needed in the future, these are already included. The processing times are given in minutes. In practice these travel times differ slightly for the different train types. For simplicity we use a fixed travel time between two machines that is equal for all train types. These travel times are based on the shortest routes, hereby assuming that tracks are long enough, so that there is always enough space to let a train unit change direction on that track. This is a reasonable assumption since we assume all train units that enter the SB are single train units. However, we make sure that train units do not need to overtake each other on their routes. An overview of the travel times can be found in Appendix A.

4 A table with the release dates (the arrival time of a job at the SB), and the due dates (the time that a job needs to depart from the SB) for all jobs. These times are all given in minutes. One day contains 1440 minutes. Most jobs arrive on day one, and leave in the night hours of day two. To cope with this, in our model, day one starts at 08:00. So, 08:00 on day one equals 0 minutes, and for example 13:00 on day one equals 300 minutes. 07:59 on day two equals 1439 minutes.

4.4 Overview, assumptions, and constraints

We can now formulate the problem as follows:

Given:

- The layout of service station *Kleine Binckhorst*,
- a set of the available machines, grouped into stages,
- an overview of jobs,
- an overview of the processing time for every operation for every job,
- a sequence in which the operations need to be processed,
- a release and due date for every job.

- the travel times between all machines at the SB, based on the shortest path.

We need to:

- Plan all operations of the jobs such that the total tardiness of all jobs is minimized. Section 4.6.2 clarifies this objective.

Such that:

- All assumptions and constraints hold, these are described in the next paragraphs.

The next paragraphs provide an overview the assumptions and constraints that are made for creating the model. The assumptions and constraints are classified in categories related to the operations, jobs, machines and SB layout.

Operation-related

Assumptions

1. There are five operations that together contain all nine first-line services that are provided at SB *Kleine Binckhorst*. The processing time of an operation is equal to the total processing times of the first-line services that, that operation consists of.
2. If a job does not require one or more first-line services, that service gets a processing time of zero, instead of its normal processing time. However, a job always contains the first and last first-line services, the arrival and departure of the SB.

Constraints

1. All operations of every job are assigned to exactly one machine.
2. Operations are processed on the machines that can process these operations.
3. All operations are processed in the defined sequence.

Machine-related

Assumptions

1. All machines are available at all times, including the internal cleaning machines.
2. All machines that can process the same operation are uniform.

Constraints

1. Every machine can process only one operation at the time.

Job-related

Constraints

1. Jobs leave the SB either at their due date, or after they have received all operations, when its completion time exceeds its due date.
2. All jobs visit all stages, even when the processing time of the operation for that stage is zero.

3. The order of visiting the stages is equal for all jobs.
4. A job cannot start being processed before its release date.

SB layout-related

Assumptions

1. The SB is empty at the start of the plan, and after the plan has ended.
2. Enough mechanics, train drivers, and other personnel that is needed to process the services, are available in excess at all times. We only included the limited cleaning personnel as is explained in the next constraint.

Constraints

1. Only one cleaning crew is available at a time, meaning that only one job can be processed on one of the two cleaning machines at the time.
2. There are no buffers between the different stages, meaning that a job keeps the machine that it is currently located on, occupied until it moves to the next machine. This means that if the next machine is still occupied by another job, the job on the previous machine has to wait before it can travel to its next machine. It also implies that a job cannot be parked between operations, so all operations are processed continuously. The processing of operations is only interrupted by travel and waiting times. Because there are no buffers, jobs cannot overtake each other if there is only one machine for that operation available.
3. Only one job can occupy a machine at the time, this means processing and waiting at the machine. However, for the calculation of the travel times between the machines it is assumed that there is always enough space for train units to use that track for traveling to the next machine. Train units do not need to pass a train unit that is being processed to reach their next machine, they need to use the track for changing directions.

The assumptions are made to simplify the model. Some assumptions are made because of the scope of our research. For example, by assuming that all personnel is available at all times, we can disregard the crew scheduling problem. Other assumptions simplify modeling the first-line services planning problem, but do compromise the solution quality much. For example, the assumption that all jobs visit all machines, even though the processing time of that operation may be zero. This is because of the carousel layout, as explained in Section 4.2. Also, the assumptions that the machines are uniform and always available, allow the model to be simplified a lot and at the same time do not compromise on the solution quality.

The operation-, machine-, job- and SB layout-related constraints are constraints that are based on the situation in practice and hold through the constraints in the model. Section 4.6.2 describes these. Section 4.6.1 describes the running example that helps in explaining the model. The description of the running example mentions all constraints that the model includes, and explains in which equations in Section 4.6.2 the constraints are displayed.

4.5 Model description

This section presents the indexes, the parameters and the decision variables that are used in the model.

Inspiration for our model is deduced from Jain and Grossmann (2001). The model they propose is a MIP model. Since we use a ILP model we need to make sure that the constrains are linear, which is not important for Jain and Grossmann (2001). In their model, jobs need to be assigned to a machine and the sequence of the jobs on the machines needs to be determined. Furthermore, the model takes into account the different release and due dates of the jobs. In our model we also include these features, but we extend it in the following ways:

- We extend the model by adding the complexity layer of operations. In our model every job consists of a set of operations. Every operation of a job needs to be assigned to a machine. The order of the operations that are assigned to the same machine, also needs to be determined. moreover, no overlap of operations on a machine may occur or overlap in processing operations of the same job at the same time.
- We extend the model of by including travel times, based on the approach used by Ahonen and de Alvarenga (2017).

Gupta (1988) proves that the flow shop problem with multiple machines at each stage is an NP-Hard problem. This also holds when one of the two stages contains only one machine. The flexible flow shop problem is a specific case of the flow shop problem. Therefore, we conclude that our problem is also a NP-Hard problem. This means that the problem cannot be solved in polynomial time. Michalewicz and Fogel (2004) explain that 'an algorithm with rational input is said to run in polynomial time if there is an integer k such that it runs in $O(n^k)$ time, where n is the input size, and all numbers intermediate computations can be stored with $O(n^k)$ bits. For an NP hard problem this is not true.

Table 4.2 provides an overview of the indexes, parameter and decision variables that are used in the model. This table is split into two parts. Part I contains the indexes and parameters and part II the decision variables.

Table 4.2: Indexes, Parameters & Decision variables (Part I)

Indexes	
o	Operation
j, v, g	Job
k_1, k_2	Machine
Parameters	
a	A job contains a operations, i.e. a is not the cumulative number of all operations of all jobs, but the last operation of every Job j. All sets of operations start at Operation 1 and end with operation a. At SB *Kleine Binckhorst* there are five operations. Therefore a equals 5, (o = 1,2,...,a).
n	Number of jobs, (j = 1,2,...,n; v = 1,2,..., n; g = 1,2,..., n).
s_o, e_o	Start and end number of machines on which operation o can be processed. In Section 4.2, we referred to these groups of machines as *stages*. Since the stages and operations correlate one to one, we model the groups of machines subject to the operations. There are in total twelve machines at SB *Kleine Binckhorst*. Operations can only be processed on certain machines, for example operation 1 can be processed on the first eight machines at the SB, and operation 2 only on machine 9. We choose to make subsets of the machines per operation, to keep the model small. This way, less options for, for example, the travel times need to be calculated in Equation 4.15 in Section 4.6.2. So, for operation 1, s_1 =1 and e_1=8; for operation 2, s_2 =9 and e_2=9; s_3 =10 and e_3=11; s_4 =12 and e_4=12; s_5 =1 and e_5=8.
R_j	Release date of job j, (j=1,2,...n).
D_j	Due date of job j, (j=1,2,...n).
M	Big-M, large integer.
$P_{j,o}$	Processing time of operation o of job j. The processing time of operation 5 is represented by the decision variable PP_j due to its dynamic nature, (o = 1,2,...,$a-1$; j=1,2,...n).
TT_{k_1,k_2}	Travel time from machine k_1 to machine k_2, (k_1, k_2=1,2,...,12)

Table 4.2: Indexes, Parameters & Decision variables (Part II)

Decision variables	
S_{j,o,k_1}	Time instant at which operation o of job j starts on machine k_1, (j=1,2,...n; o = 1,2,...,a; $s_o \leq k_1 \leq e_o$).
$S_{j,o}$	Time instant at which operation o of job j starts, (j=1,2,...n; o = 1,2,...,a).
$C_{j,o}$	Completion time of operation o of job j, (j=1,2,...n; o = 1,2,...,$a-1$).
C_j	Completion time of the last operation o of job j, (j=1,2,...n).
PP_j	Processing time of the last operation of job j. This processing time is not fixed for job j, because it depends on the arrival time of the job on the lasts machine, and the job's due date. This is because the job cannot leave the SB before its due date. We explain this calculation in more detail in Equations 4.19 in Section 4.6.1, (j=1,2,...,n).
$W_{j,o}$	Waiting time after operation o of job j is processed, and before job j can start to travel to its next machine on which operation $o+1$ will be processed, (j=1,2,...n; o = 1,2,...,$a-1$).
$DT_{j,o}$	Duration of operation o of job j. Equation 4.9 explains this decision variable in more detail, (j=1,2,...n; o = 1,2,...,$a-1$).
TJ_{j,o,k_1}	The summation of the travel time of operation o of job j, from machine k_1 to all machines on which operation $o+1$ can be processed, (j=1,2,...n; o = 1,2,...,$a-1$; $s_o \leq k_1 \leq e_o$).
$TTO_{j,o}$	The eventual travel time from one machine k_1 to the next machine k_2 after operation o of job j. The travel times are explained in more detail in Equations 4.15 and 4.16, (o = 1,2,...,$a-1$).
$Tardiness_j$	Tardiness of job j, (j=1,2,...,n).

Equations 4.2, 4.3, and 4.4 display the binary decision variables

$$Y_{j,o,k_1} = \begin{cases} 1, & \text{if operation } o \text{ of job } j \text{ is processed} \\ & \text{on machine } k_1 \\ 0, & \text{otherwise} \end{cases} \tag{4.2}$$

$\forall j, o$ with: $s_o \leq k_1 \leq e_o$.

$$X_{j,v,k_1} = \begin{cases} 1, & \text{if job } j \text{ precedes job } v \text{ on machine } k_1 \\ 0, & \text{otherwise} \end{cases} \tag{4.3}$$

$\forall j \neq v;\ k_1 = 1, 2, ..., 12.$

$$U_{j,h} = \begin{cases} 1, & \text{if the sub-equation of Equation 4.17,} \\ & \text{does not need to be binding for job } j \\ 0, & \text{otherwise} \end{cases} \tag{4.4}$$

$\forall j;\ h = 1, 2, 3, 4,$ explanation follows in Equation 4.17 and Equation 4.18.

In Section 4.6 we describe the constraints of the model.

4.6 Mathematical model

This section provides the constraints and definitions that are used our ILP model. The model is explained using a running example. Subsection 4.6.1 describes the example, and Section 4.6.2 explains the constraints and definitions of the model. Section 4.7 provides some conclusions of the model.

4.6.1 Running example

In this running example there are two jobs, so n = 2. Table 4.3 provides an overview of the characteristics of the two jobs. Both jobs are of a train type, and contain a set of required first-line services that are needed for the processing times of the operations for job 1 and job 2. The calculation of the processing times of the operations is not described in the model, because we use these processing times as input. The processing times are simply determined by adding the processing times of the required first-line services of job j's train type. Furthermore, the release and due dates for the jobs are displayed in Table 4.3.

Table 4.3: Running Example characteristics

Job(j)	Train type	Required first-line services	Release date(R_j) minutes (time instant)	Due date(D_j) minutes (time instant)
1	SLT4	1,3,6,9	1081 (02:01)	1250 (04:50)
2	VIRM4	1,2,6,8,9	1095 (02:15)	1282 (05:21)

Table 4.4 displays the processing times for the five operations of each job. For job 1, a processing time of zero is assigned to first-line service 8. Therefore, operation 4 also has a processing time of zero. However, as said before, job 1 still has to visit machine 12, the only machine that can process operation 4, because of the carousel layout.

Table 4.4: Processing time of the operations of job 1 & 2

Job(j)	Operation(o)	Processing time
1	1	49
1	2	60
1	3	73
1	4	0
1	5	$max(0, C_{j,4} - D_j)$
2	1	21
2	2	30
2	3	186
2	4	24
2	5	$max(0, C_{j,4} - D_j)$

The processing time of the last operation is dynamic. The processing time of the fifth operation is the time between the job's due date, and its arrival on the job's last machine. When the arrival time of the job on its last machine exceeds the job's due date, the processing time of the last operation equals zero, and the job leaves the SB directly.

Figure 4.3 displays a possible outcome of the running example. This feasible solution displays all aspects for which the model includes constraints. In this Gantt chart, the y-axis displays the numbers of the machines on which the operations of the jobs are planned, and the x-axis represents the time. The green blocks represent the operations of job 1, and the blue blocks represent the operations of job 2.

The next section explains the Gantt chart step by step. It clarifies all constraints and definitions that the model includes. The figure displays every decision variable at most once, even though they are present more often. For example, we display the start time of an operation for both jobs only once, but naturally all operations have a start time. Figure 4.3 provides short explanations about the displayed decision variable. All these explanations have a different capital letter to distinguish them.

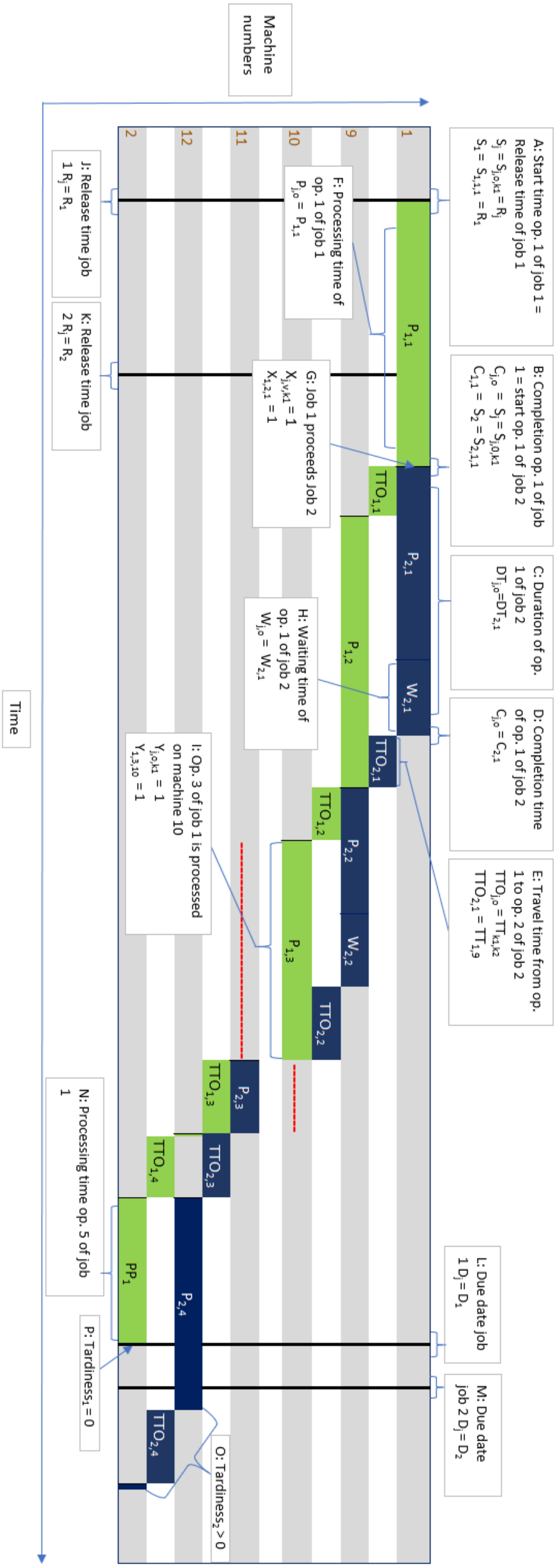

Figure 4.3: Example jobs planned

4.6.2 Model

This section describes the model by using the running example as described in the previous section. The Gannt chart in Figure 4.3 guides the explanation. We describe all the constraints and definitions that the model includes and display the constraint as included in the model.

Both jobs have a release date R_j, and a due date D_j, appointed by the letters *L* & *M* and *J* & *K* in Figure 4.3. The start time of the first operation of a job S_j, needs to be greater than, or equal to the release date of job j. We allow the jobs to start their operations later than their release date, because in our model every machine can only contain one job. In practice however, more train units can be parked on the stabling tracks. Therefore, the jobs that start their operations later than their release date, are assumed to be parked on the assigned track for their arrival operation, from their release date on. Equation 4.5 ensures this. In our example, operation 1 of job 1 cannot start before 1081 minutes and operation 1 of job 2 needs to start at, or after 1095 minutes.

$$R_j \leq S_{j,1} \qquad \forall j. \tag{4.5}$$

For job 1, the start time of the first operation is equal to the release date. This operation is processed on machine 1, so the start time of the first operation of the first job on machine 1 is denoted by $S_{1,1,1}$. Equation 4.6 determines the value of $S_{1,1,1}$

$$S_{j,o,k_1} \leq 1440 * Y_{j,o,k_1} \tag{4.6a}$$

$$S_{j,o,k_1} \leq S_{j,o} \tag{4.6b}$$

$$S_{j,o,k_1} \geq S_{j,o} - 1440 * (1 - Y_{j,o,k_1}) \tag{4.6c}$$

$$\forall j \neq v;\ \forall o;\ s_o \leq k_1 \leq e_o.$$

The three sub-equations displayed in Equation 4.6 determine the start time of operation 1 of job 1 on machine 1. Equations 4.6a and 4.6b create bounds for Equation 4.6c. If Y_{j,o,k_1} is zero, Equation 4.6a ensures S_{j,o,k_1} becomes zero, because of Equation 4.6a and the domain of S_{j,o,k_1}, displayed in Equation 4.23. If Y_{j,o,k_1} equals one, Equation 4.6b ensures S_{j,o,k_1} equals $S_{j,o}$. This is how the machine-independent start times $S_{j,o}$ are linked to the machine k_1 , on which operation o of job j is processed.

All operations of all jobs are assigned to exactly one machine that may process that operation. This is ensured by Equation 4.7, and displayed by the letter 'I' in Figure 4.3.

$$\sum_{k_1=s_o}^{e_o} Y_{j,o,k_1} = 1 \qquad \forall j, o. \tag{4.7}$$

The first green block displays the processing time of the first operation of the first job: $P_{1,1}$, denoted by *F* in Figure 4.3 . The completion time of operation 1 of job 1 is denoted by $C_{1,1}$, displayed by the letter *D* in the figure and shown in Equation 4.8. The completion time of operation o is the sum of the start time of operation o, and its duration. The duration of an operation is the sum of the processing time and the waiting time of operation o of job j. This is in Figure 4.3 displayed at letter *C*. Equation 4.9 displays the calculation of the duration. We explain the waiting time later in Equation 4.11.

$$C_{j,o} = S_{j,o} + DT_{j,o} \qquad \forall j; o = 1, 2, ..., a-1. \tag{4.8}$$

$$DT_{j,o} = P_{j,o} + W_{j,o} \qquad \forall j; o = 1, 2, ..., a-1. \tag{4.9}$$

After operation 1 of job 1 is completed, job 1 travels to machine 9 and operation 1 of job 2 can start processing on machine 1, denoted by $S_{2,1,1}$.

Since we model the first-line services problem as a flexible flow shop, the sequence in which all operations of the jobs are processed is fixed. This fixed sequence is ensured by Equation 4.10. The operations are processed in increasing order. All start times of operation $o+1$ are greater than the completion time of operation o. This way operation 1 of job 1 is processed before operation 2 of job 1. Also, this equation ensures that all stages with machines are visited, even though the processing time of operation o might be zero. $TTO_{j,o}$ is the travel time between the machine that processes operation o, and the machine that processes operation $o+1$. How we select this travel time, we explain later in Equations 4.15 and 4.16.

$$\begin{gathered} C_{j,o} + TTO_{j,o} = S_{j,o+1} \\ \forall j;\ o = 1, 2, ..., a-1. \end{gathered} \tag{4.10}$$

Operation 2 of job 1 can start being processed on machine 9 as soon as the job arrives on the machine. The letter *H* in Figure 4.3 shows that the processing time of operation 2 of job 1 exceeds the completion time of operation 1 of job 2. Job 2 needs to wait on machine 1, because machine 9 is still occupied and there are no buffers at SB *Kleine Binckhorst* . The waiting time is calculated as displayed in Equation 4.11

The waiting time is the time between the start time of operation $o+1$ on machine k_2, and the completion time of operation o on machine k_1, minus the travel time between machine k_1 and k_2. In our example, job 2 can start on machine 9 as soon as job 1 leaves. Therefore, job 2's waiting time after completing operation 1 is the time until job 2 can start being processed on machine 9, minus its travel time to machine 9.

$$W_{j,o} \geq S_{j,o+1} - C_{j,o} - TTO_{j,o} \quad \forall j;\ o = 1, 2, ...a - 1. \tag{4.11}$$

Job 2 needs to wait on machine 1 because machine 9 is occupied. That only one operation can be processed on one machine is ensured by Equations 4.12 and 4.13.

In our example both jobs need to be processed on machine 9. So, a processing order needs to be determined. Either, job 1 precedes job 2, or the other way around. Assume, job 1 is job j and job 2 is job v. If job 1 precedes job 2, $X_{1,2,9}$ equals one, as defined in Equation 4.3, and Equation 4.12a is binding. In this case, the start time of job v needs to be larger than the start time and duration of job j. This way there is no overlap of the two operations on machine 9. Equation 4.12b still holds, but is not a limiting constraint. If job 2 would precede job 1, Equation 4.12b would be the binding constraint, and Equation 4.12a would hold, but not be binding.

$$S_{v,o,k_1} \geq S_{j,o,k_1} + DT_{j,o} - M * (1 - X_{j,v,k_1}) - M * (1 - Y_{j,o,k_1}) - M * (1 - Y_{v,o,k_1}) \tag{4.12a}$$

$$S_{j,o,k_1} \geq S_{v,o,k_1} + DT_{v,o} - M * X_{j,v,k_1} - M * (1 - Y_{j,o,k_1}) - M * (1 - Y_{v,o,k_1}) \tag{4.12b}$$

$$\forall j \neq v;\ \forall o; s_o \leq k_1 \leq e_o$$

The big-M value enables only one of the two sub-equations to be binding. To make sure that we do not expand the solution space unnecessary, to enable the model to be solved quickly, we need to choose a large value of big-M, yet, this value needs to be as small as possible (Camm et al., 1990). We choose the value for big-M to be 1511. This is the sum of latest start time, as determined in Equation 4.6, which is 1440 minutes, and the longest processing time of the fourth operation, which is 111 minutes. This time will never be exceeded, because the model cannot start operations after 1440 minutes. Therefore, no feasible solution exists for the planning problem if an operation of a job cannot start before 1440 minutes. We use this value for big-M in all equations.

Since operation 1 and operation 5 share the same set of machines on which these operations can be processed, we need Equation 4.13. This is to make sure operation 1 and operation 5 do not overlap on the same machine.

$$S_{v,1,k_1} \geq S_{j,5,k_1} + DT_{j,5} - M * (1 - X_{j,v,k_1}) - M * (1 - Y_{j,5,k_1}) - M * (1 - Y_{v,1,k_1}) \tag{4.13a}$$

$$S_{j,5,k_1} \geq S_{v,1,k_1} + DT_{v,1} - M * X_{j,v,k_1} - \\ M * (1 - Y_{j,5,k_1}) - M * (1 - Y_{v,1,k_1}) \tag{4.13b}$$

$$\forall j \neq v;\ k_1 = 1, 2, ..., 8$$

For Equation 4.12 and 4.13 the X-variable is important, because this variable determines the sequence of the jobs on the machine. In other words, either job 1 precedes job 2, or job 2 precedes job 1. In our example, job 1 precedes job 2, so, $X_{1,2,1}$=1. This is displayed at letter *G* in Figure 4.3.

Equation 4.14 displays the other important constraint to ensure that job 2 waits on machine 1. This constraint ensures that a job cannot move to its next machine, until that machine is available, minus the job's travel time to that machine. This is, as just explained, because there are no buffers at SB *Kleine Binckhorst*.

Again, assume that job 1 is job j and job 2 is job v. In our example, job j precedes job v on machine 9. This means that Equation 4.14 needs to makes sure that operation o of job j cannot start before job v has left machine k_1. So, in our example, job 2 cannot start being processed on machine 9, before job 1 starts being processed on machine 10, minus job 1's the travel time to machine 9.

$$S_{j,o,k_1} \geq S_{v,o+1,k_2} - TT_{k_1,k_2} - M * (1 - X_{j,v,k_1}) - \\ M * (1 - Y_{j,o,k_1}) - M * (1 - Y_{v,o,k_2}) \\ \forall j \neq v;\ o = 1, 2, ..., a-1;; \\ s_o \leq k_1 \leq e_o \tag{4.14}$$

When an operation of a job is completed, the job travels to a next machine. To select the right travel time between the machines that the job travels from and to, we need Equations 4.15 and 4.16. These equations calculate $TTO_{j,o}$, which is displayed at letter *E* in Figure 4.3. The travel times are provided by a parameter, however, since it is up to the model to decide on which machines the operations of a job are processed, we need these two equations to use the travel time between the right two machines.

First, as Equation 4.15 displays, we select the travel times between all machines that can process operation o and all machines that can process operation $o + 1$. In our example, operation 2 of job 1 is processed on machine 9 and operation 3 on machine 10, but operation 3 could also be processed on machine 11. So, to select the right travel time, we add the travel times between machine 9 and machine 10, and machine 9 and machine 11. The travel times between machines on which the operation is not processed, equal zero, because of the Y-variables that are zero if the operation is not processed on that machine, and because of the domain of TJ_{j,o,k_1} displayed in Equation 4.23. Only the travel time between the two machines on which the operations are processed has a value greater than zero. In our example, the travel time between machine 9 and machine 11 equals zero, and the travel time between machine 9 and 10 equals the real travel time. So, $TJ_{1,2,9}$ equals the travel time from machine 9 to machine 10.

In our example, operation 2 can only be processed on machine 9, if there was another machine on which operation 2 could be processed, we need the next step. Assume, only for this explanation, that operation 2 can also be processed on machine 8. In this case, in the previous step, we would have calculated the values for $TJ_{1,2,8}$ and $TJ_{1,2,9}$. One of the TJ-variables contains the real travel time, and the other one equals zero, again, because of the Y-variables used in Equation 4.15. Therefore, we need to add these two TJ values together to find the real travel time. This happens in Equation 4.16.So, $TTO_{1,2}$ represents the right travel time.

This calculation might seem cumbersome, but, because we want to keep our model linear we need to determine the required travel times through these two steps.

$$TJ_{j,o,k_1} \geq \sum_{k_2=s_{(o+1)}}^{e_{(o+1)}} TT_{k_1,k_2} * (Y_{j,o,k_1} + Y_{j,o+1,k_2} - 1) \qquad \forall j;\ o = 1,2,...,a-1;\ s_o \leq k_1 \leq e_o; \tag{4.15}$$

$$TTO_{j,o} = \sum_{k_1=s_o}^{e_o} TJ_{j,o,k_1} \qquad \forall j;\ o = 1,2,...,a-1 \tag{4.16}$$

Another constraint that we incorporate, is that only one cleaning crew is available to process jobs on machine 10 and 11. As can be seen in Figure 4.3 at letter *I*, during the time that operation 3 of job 1 is processed on machine 10, no operation can be processed on machine 11. The dotted line on machine 11 displays the 'occupation' of the other machine. This also holds the other way around, during the time that operation 3 of job 2 is processed on machine 11, no operation can be processed on machine 10. Equations 4.17 and 4.18 describe how we incorporate this in our model.

Equation 4.17 ensures that only one job's operation 3 on machine 10 or 11 is treated at the time. Independently of the number of jobs that needs to be planned, there are four situations that need to be considered. Either job 1 on machine 10 precedes job 2 on machine 11, or the other way around. Or job 1 on machine 11 precedes job 2 on machine 10, or the other way around. Again, the assignment of the jobs to the machines is done by the model. Therefore, all four sub-equations are needed but only one of the four needs to be binding. Equation 4.18 ensures this by allowing only one of the four U

variables to equal 1. If machine k_1 = 10 and k_2 = 11, then, for our example, Equation 4.17a needs to hold, because job 1 on machine 10 precedes job 2 on machine 11.

$$\begin{aligned} S_{j,3,k_1} \geq S_{g,3,k_2} + P_{j,o} - M * (1 - Y_{j,3,k_1}) \\ -M * (1 - Y_{g,3,k_2}) - M * U(j,1) \end{aligned} \tag{4.17a}$$

$$\begin{aligned} S_{g,3,k_1} \geq S_{j,3,k_2} + P_{j,3} - M * (1 - Y_{g,3,k_1}) \\ -M * (1 - Y_{j,3,k_2}) - M * U(j,2) \end{aligned} \tag{4.17b}$$

$$\begin{aligned} S_{j,3,k_2} \geq RS_{g,3,k_1} + P_{j,3} - M * (1 - Y_{j,3,k_2}) \\ -M * (1 - Y_{g,3,k_1}) - M * U(j,3) \end{aligned} \tag{4.17c}$$

$$\begin{aligned} S_{g,3,k_2} \geq RS_{j,3,k_1} + P_{j,3} - M * (1 - Y_{g,3,k_2}) \\ -M * (1 - Y_{j,3,k_1}) - M * U(j,4) \end{aligned} \tag{4.17d}$$

$$\forall j \neq g;\ k_1, k_2 = 10, 11 \text{ with } k_1 \neq k_2$$

$$U(j,1) + U(j,2) + U(j,3) + U(j,4) = 1 \qquad \forall j \tag{4.18}$$

As described before, the processing time of operation 5 is dynamic. The calculation of this processing time is described by Equation 4.19 and displayed at letter 'N'. It is the time between job j's arrival on its last machine and its due date. If job j arrives on its last machine after its due date, PP_j equals zero, ensured by the domain constraint in Equation 4.23. To calculate PP_j as displayed in Equation 4.19, we need to include the completion time of operation 4, because the $TTO_{j,4}$ represents a travel time between machines, and not a moment in time. If the due date of the job is not exceeded yet, operation 5 has a duration greater than zero. This is displayed in Figure 4.3 by job 1. Job 2 already exceeds its due date during the processing of operation 4. Therefore PP_2 equals 0.

$$PP_j \geq D_j - (C_{j,4} + TTO_{j,4}) \quad \forall j. \tag{4.19}$$

Job j's final completion time is denoted by C_j and displayed in Equation 4.20. It is very similar to Equation 4.8, but this equation only holds for the last operation of job j. This completion time is used in the objective.

$$C_j = S_{j,5} + PP_j \qquad \forall j \tag{4.20}$$

Finally, the objective of our model is to minimize the total tardiness of all jobs. The tardiness of a job is the time that the completion time of a job exceeds its due date. We minimize the tardiness to make sure the total time that the train units are delayed, is kept as little as possible. Equation 4.21 displays the calculation of the tardiness, which is the time between the completion time of a job, and its due date.

$$Tardiness_j \geq C_j - D_j \qquad \forall j \tag{4.21}$$

Equation 4.22 displays the objective. In our example, job 2 is delayed, and thus has a positive tardiness, shown at the letter 'O'. For job 1, the tardiness is zero, as displayed at the letter 'P'.

$$Min \sum_{j=1}^{n} Tardiness_j \tag{4.22}$$

For all decision variables the following domains hold:

$$S_{j,o,k_1},\ S_{j,o},\ C_{j,o},\ C_j, \geq R_j \tag{4.23a}$$
$$PP_j,\ W_{j,o},\ DT_{j,o},\ TJ_{j,o,k_1},\ TTO_{j,o},\ Tardiness_j \geq 0 \tag{4.23b}$$
$$Y_{j,o,k_1},\ X_{j,v,k_1},\ U_{j,h} \in [0,1] \tag{4.23c}$$
$$\forall j, o, h, k_1$$

4.7 Model conclusion

To conclude on this chapter we provide a small summary of the model and provide remarks on the running example.

The model that we develop to solve the first-line service planning problem for the NS service station *Kleine Binckhorst* is an ILP model, because all constraints are linear and all values of the parameters and decision variables integer. In the model, the nine first-line services are aggregated into five operations. The sequence in which the jobs process all operations is equal and fixed.

The model includes constraints to make sure that simple aspects of the model hold. For example only one operation can be processed on one machine at the time. However, some more special features are also included in the model, such as the no-buffer, waiting and and sequencing features. Also, we include travel times between machines. We leave it up to the machine to determine which operation to process on which machine and in which sequence. Therefore, the right travel distance needs to be selected. Finally, we include recirculation in our model, resulting in capacity sharing of one set of machines.

In comparison with practice most important features and constraints of SB *Kleine Binckhorst* are included in our algorithm to model the reality as truthful as possible. Algorithms developed by other scholars often include only a few of the features that we include in our algorithm. For example, Ahonen and de Alvarenga (2017) create a algorithm that assigns operations of jobs to machines, includes travel times and recirculation. However, they have unlimited buffer capacity between the machines. We also include the constraints that jobs need to wait on their current machine until their next machine is empty.

To explain all the constraints that are included in the model, we discuss a running example. Figure 4.3 is one possible solution, it is not per sé the t solution. For example, the model would probably not process the first operations of the jobs on the same machine since there are eight machines on which this operations is allowed to be processed. However, the next machine is machine 9, the only machine that can process operation 2 of both jobs, so job 2 still needs to wait on job 1 to leave the machine.

5 | Results

This chapter describes the results of the model and answers the fourth subquestion: *For which problem instances can we find a first-line services plan, using the model?* To test the model, we provide insights in the internal cleaning capacity of the SB, and on the capacity of the model. With the internal cleaning capacity of the SB, we refer to the number of train units that need to be internally cleaned for which we can find a feasible plan. Section 5.1 elaborates on how we do this. Section 5.2 extends the results of Section 5.1 including more first-line services in the planning.

In reality up to 20 train units enter SB *Kleine Binckhorst* every night, and about eight during day time. These train units require a set of first-line services to be executed during their stay at the SB. Section 5.1 describes Experiment 1, in which we consider three first-line services in the first-line services plan. Section 5.2 describes the second experiment. This experiment includes a realistic set of first-line services per job.

We use the same the test cases for both experiments. These are generated by NS and provide realistic arrival and due dates, and train types of the train units. Appendix B describes how we modify the data to fit our model. We solve different test cases that start with two train units, and increase per two train units. Every test case consists of ten problem instances in which the train units have different release and due dates, but every problem instance includes the same train types and number of train units of this train type. In practice a train unit is only internally cleaned if it stays on the SB for at least two hours. We obey this constraint and make sure that there is at least two hours between the release and due date of the train unit. Moreover, the carousel layout still holds, so the jobs need to visit all five assigned machines.

We solve the problem instances using CPLEX 12.8 on a Intel core i5 - 3337U 1.8 GHz computer with 3.4 Gbyte of RAM memory.

5.1 Experiment 1

At SB *Kleine Binckhorst*, the internal cleaning machines are regarded as the planning bottleneck. Cleaning a train unit internally is very important, because NS wants to deliver a high service level to its customer, and a clean train is a priority in this. Therefore, the train units need to be cleaned every day, see Table 5.1. This means that, basically, all train units that enter the SB require this first-line service. NS does not have an estimation of the capacity of internal cleaning machines at SB *Kleine Binckhorst* yet. If the internal cleaning station is occupied constantly, there are daily 1440 minutes available for internal cleaning. This is because only one cleaning crew is available, thus one internal cleaning machine can process a train unit at a time.

We refer to the capacity of the internal cleaning machines as the number of train units for which a feasible plan can be found, within realistic release and due dates. To estimate this, we solve the by NS generated problem instances. This way we provide insights in the number of train units for which it is likely a feasible plan can be found. The problem instances that we solve only plan the arrival, internal cleaning and departure first-line services.

Table 5.1: First-line services requirement periodicity for a train unit

First-line service	interval (*days*)
Arrival	1
InspectionB	2
InspectionA	12
Telehandler	30
Workpit	30
Internal cleaning	1
External cleaning normal	7
External cleaning Oxalic	63
Departure	1

In this experiment, we analyze the problem instances on three aspects:

- Tardiness: Tardiness is the objective of the model that we want to minimize, such that the train units can leave as much on time as possible.
- Runtime of the model: The problem instances have a run time limit of two hours, but ideally, the problem instances are solved within 5 minutes.
- Status of found plan: We asses the found plan on if the plan is optimal, feasible, or infeasible.

We analyze the capacity of the SB for problem instances $2 \leq n \leq 16$ jobs. This is because the problem instances of the test case with $n \geq 18$ cannot be solved without a positive tardiness. Plans that include train units with a positive tardiness cannot be used by NS. Release and due dates of train units are fixed, so train units cannot be delayed. The train units' release and due dates at an SB are set by the passenger transport time table, and need to be obeyed by the SB. This because the passenger train schedule is leading in the whole rolling stock planning. So, there is no room in changing the arrival and departure times of train units, to improve the first-line services planning at the SB. The test cases that we use include realistic release and due dates for the train units. We do not change these to see what the impact is on the solution because in reality there also is no room in changing these. The only way to changing a plan that has a positive total tardiness, is by removing some internal cleaning first-line services for certain train units.

Tables 5.3 and 5.4 show the results of the experiment. Appendix D shows the individual results of the problem instances for test case $n = 2, 4, 6, \&8$. The first column the number of train units in the problem instances. Subsequently, Column 3 displays the total tardiness and Column 5 displays the maximum tardiness for one job within the problem instance. The gap and lower bound are provided if the solution is not optimal. In this case the objective is the best solution found so far. Most problem instances have a tardiness of zero. Of the ten problem instances with $n = 14$, two problem instances have a positive tardiness, and of the problem instances with $n = 16$ five problem instances result in a positive tardiness. We analyze the positive tardiness for the fifth problem instance of $n = 14$ later in this section.

The second aspect that we analyze the results on, is the run time of the model. Problem instances with $n = 2$ and $n = 4$ are solved in less than one second. Problem instances with $n = 6, 8, 10$ are solved between 4 and 16 seconds. The more n increases, the more variable the runtime of the model becomes. This has to do with the size of the problem instance. For some problem instances, quickly large branches can be disregarded, when the relaxed solution is solved and the model tries to find an integer solution. For other problem instances more time is needed. We see this in extreme for $n = 16$, here one problem instance can be solved to optimality in less than 74 seconds and for other problem instances two hours is not enough to solve the problem instances to optimality. Finally, for the ninth problem instance for which $n = 14$, no feasible solution exists. The model concludes this very quickly. This is because the model can probably not even solve the relaxation of the model. The reason for this infeasibility is that operations need to start after 1440 minutes to create a plan. This is not allowed in our model, because this causes tardiness for the job and thus problem instance.

The third aspect is the solution status. All problem instances that are solved within two hours are solved to optimality. The problem instances that cannot be solved within two hours result in a feasible solution. Only for the ninth problem instance for n-14, no solution is found at all.

Concluding, many solutions are found within a few seconds. This is a positive result. NS develops these planning models to assist the planners in decision making. For this small run times are required because it is not possible to run a model two hours for every change that happens and quick decision making is required. Problem instances up to 16 train units can be planned that include internal cleaning. In reality up to 28 train units enter the SB. However, it is not clear how many of these are actually internally cleaned every day.

Examining the fifth problem instance of test case $n = 14$ in more detail, we see that two jobs experience tardiness. Table 5.2 displays the tardiness of the fourteen jobs in the second column. The other columns show the release dates of the jobs and the start time of the first operation which we need later in this section. In Figure 5.1 the y-axis displays the 14 jobs and the x-axis represents the time, starting at 0 minutes and ending at 1500 minutes. The colored blocks in the figure represent the processing time of the operations of the jobs that are planned. The blank spaces between the colored blocks display the travel times. The first and third operation are displayed for all jobs and the fifth operation for only seven jobs. This means that the other jobs arrive on their last machine, on their due date. These jobs have a processing time of zero for the last operation.

Figure 5.2 displays on the y-axis the 12 machines of SB *Kleine Binckhorst* and the x-axis represents the time. Again, the time starts at 0 minutes and ends at 1500 minutes. In this figure, the colored blocks represent the duration of the activities that are planned. Recall, the duration of an activity is the sum of the processing time of that activity and the time the job waits on its current machine after its activity is completed. All operations are displayed in the figure, even though only the first, third operation have a positive processing time, and the last operation's processing time is variable. So, the second, and fourth operation's duration displays the job's waiting time on that machine.

Analyzing the cause of the tardiness of the two jobs, we see that the tardiness of job 9 is caused by too tight release and due dates, because the job experiences no waiting time. Job 8 also experiences tardiness, this is because the model chooses to process operation 2 of job 9 first, even though job 8 enters the SB earlier. Therefore, job 8 needs to wait on job 9. In the model we minimize the total tardiness, thus the total tardiness of planning the operations is less in this plan, than when the jobs would be planned according to the first in, first out principle. Table 5.2 shows that job 5 has a late start. This is allowed by Equation 4.5 in Section 4.6.2 and is done because if we would have implemented a dynamic duration of the arrival activity the tracks would be occupied to long and the solution quality would be less. In practice, multiple train unit can occupy the same track. However, in our model only one train unit can occupy a track. This was the most practical to deal with this issue.

We conclude from this experiment that problem instances for which $2 \leq n \leq 12$ can be planned within seconds, with a tardiness of zero. For problem instances with $12 \leq n \leq 16$ not all problem instances can be planned without a positive tardiness. Therefore, we conclude that the capacity of SB *Kleine Binckhorst* is limited by 16 train units that need to be internally cleaned. This tardiness can be due to waiting times caused by the internal cleaning station on which only one job can be processed at a time. The other cause is tight release and due dates. We obeyed the assumption that a job needs to be at the SB for at least two hours. The processing time of cleaning activity is different per train unit type, so these two hours might not be enough for every train unit.

Table 5.2: Important data of fifth problem instance, n=14

Job	Tardiness	Release date	Start time first operation
1		621	706
2		887	1023
3		887	926
4		643	643
5		1113	1168
6		600	600
7		643	643
8	6	1113	1113
9	7	1122	1122
10		106	106
11		176	205
12		176	205
13		874	874
14		874	874

Table 5.3: Results Experiment 1 test cases n = 12,14,16,18

		Objective for all problem instances	Run time interval
Test case	n=2	0	0.05-0.19
	n=4	0	0.21-0.63
	n=6	0	1.03-2.09
	n=8	0	2.11-8.94

Table 5.4: Results Experiment 1 test cases n = 12,14,16,18

		Obj. (min)	Gap (%) & [LB]	Job's max tardiness (min)	Time (s)
n=10	1				7.97
	2				10.16
	3				8.81
	4				9.16
	5				16.59
	6				7.67
	7				8.50
	8				12.58
	9				9.61
	10				10.95
n=12	1				418.52
	2				24.61
	3				6.23
	4				25.52
	5				11.52
	6				14.31
	7				16.96
	8				12.84
	9				12.86
	10				23.48
n=14	1				4073.63
	2				16.05
	3				4.17
	4				8.44
	5	13	0%	7	60.75
	6				4.67
	7	22	0%	22	9.50
	8				4.16
	9	Infeasible	-	-	0.06
	10				3.91
n=16	1	77	72.73%, [21]	43	7200.23
	2				273.41
	3				486.86
	4				148.55
	5				89.55
	6	63	100%	49	7200.88
	7				73.95
	8	38	100%	24	7200.42
	9	100	100%	80	7200.28
	10	57	70.18% [17]	43	7200.77

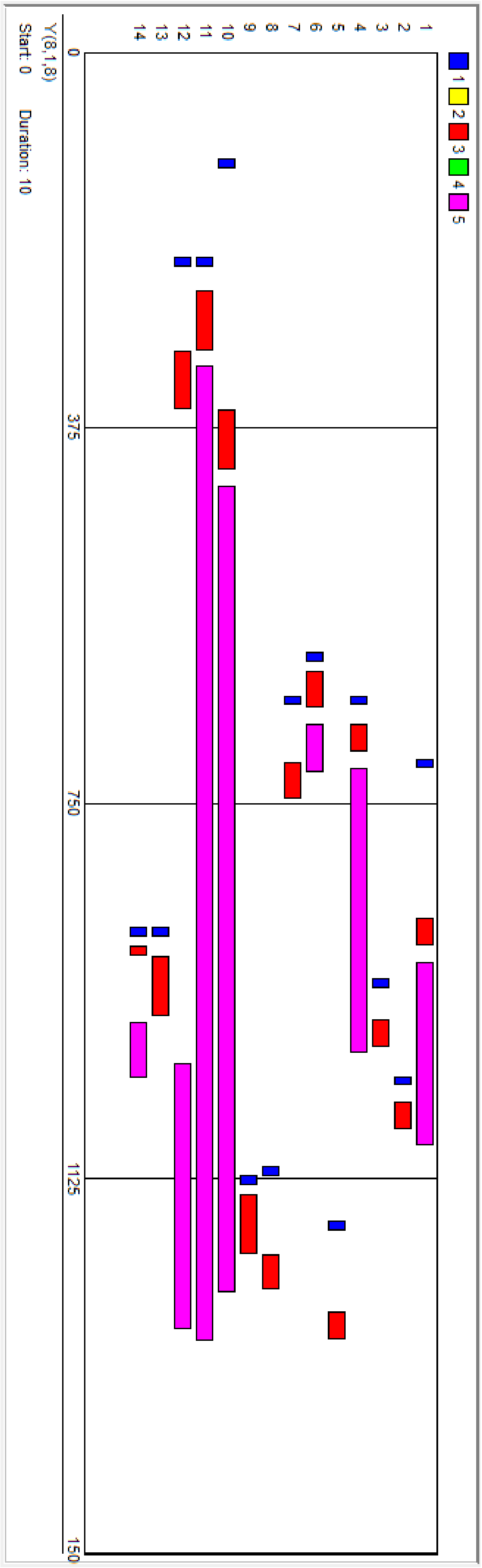

Figure 5.1: Plan problem instance 14-5 Part 1

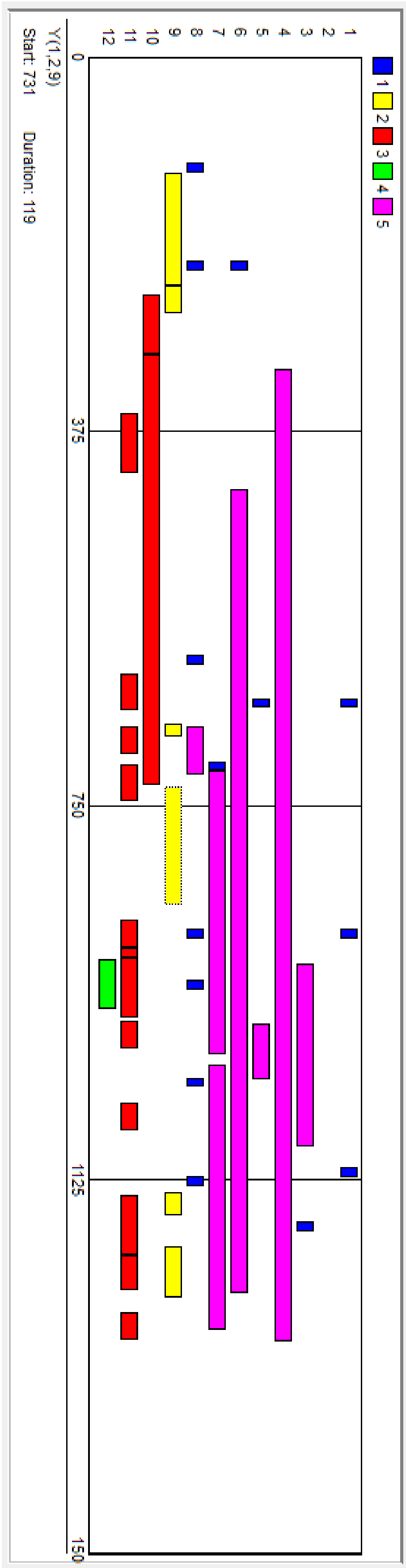

Figure 5.2: Plan problem instance 14-5 Part 1

5.2 Experiment 2

Experiment 1 focuses on the internal cleaning capacity of SB *Kleine Binckhorst*. In this experiment we also use the test cases for which $2 \leq n \leq 16$ and a runtime of maximum two hours. In this experiment, we add first-line services to jobs based on Table 5.1, in Section 5.1 to create realistic sets of required first-line services for all jobs. For example, each train unit needs to enter the SB, so every train unit gets a positive processing time for the arrival first line service. Train units require on average every twelve days an inspection A, so one in twelve train units gets a processing time greater than zero assigned for this first-line service. We round up this number, so in every set of problem instances, at least one train unit has a positive processing time for this first line service. This also holds for the other first line services. Appendix C shows the assigned first-line services for all train units.

Tables 5.5 and 5.6 show the results of this experiment. These tables have the same setup as in Experiment 1. Appendix D shows the individual results of the problem instances for test case $n = 2, 4, 6, \&8$. We see that more problem instances result in a positive tardiness in this experiment than in Experiment 1. More operations have a positive processing time, which increases the total processing time of a train unit. This influences the waiting times of the train units succeeding this train unit. Now train units cannot directly pass the machine, but might have to wait before they can be processed or pass the machine to travel to their next machine.

In this experiment, the first problem instance that results in a positive tardiness is problem instance 10-1. In Experiment 1, this problem instance did not have a positive tardiness. Job 1 experiences the tardiness in this problem instance and the tardiness is due to the increased processing times of the job's operations. Figure 5.3 shows on the y-axis the 10 jobs and displays on the first row job 1. Figure 5.4 shows the waiting time for all jobs. We conclude that the little time between the release and due dates of job 1 causes the tardiness, because Figure 5.4 shows no waiting time for job 1. All the other jobs are completed either just on time, or far in advance, such as job 3.

The first problem instance of $n = 12$ has a total tardiness of 58 divided over four jobs. Figure 5.7 shows the tardiness of the twelve jobs Figures 5.5, 5.6 show the optimal plan for this problem instance. Figure 5.5 shows the Gantt chart for the 12 machines. In Figure 5.6, the 12 jobs are displayed on the Y-axis. The tardiness of three of the jobs is due to waiting time before the jobs can start their internal cleaning operation. The fourth job experiences tardiness because of tight release and due dates.

To conclude, when adding more operations to a job, the total time that is needed for the job to process all operations and travel over the SB increases. This results more often than in Experiment 1 in a positive tardiness for problem instances. Also in this experiment the tardiness is either due to too little time at the SB for a job or because the job needs to wait on another job.

Table 5.5: Summary results Experiment 2 test cases n = 2,4,6,8

		Objective for all problem instances	Run time interval
Test case	n=2	0	0.06-0.09
	n=4	0	0.17-0.25
	n=6	0	0.22-0.66
	n=8	0	0.45-1.30

Table 5.6: Results Experiment 2 test cases n = 12,14,16,18

		Obj. (min)	gap % & [LB]	Job's max tardiness (min)	Time (s)
n=10	1	8		8	8.17
	2				0.78
	3				1.81
	4				3.34
	5				1.72
	6				3.59
	7				6.44
	8				3.03
	9				2.14
	10				1.75
n=12	1				29.70
	2				1.14
	3				4.25
	4				4.25
	5				4.26
	6				6.09
	7				4.86
	8	60		60	3.56
	9				2.03
	10	19		19	4.97
n=14	1				236
	2				9.36
	3				6.92
	4				8.13
	5	28		28	42.94
	6				3.17
	7	22		22	3.34
	8				2.78
	9	infeasible		-	0.05
	10				6.83
n=16	1	77	72.73% 21	36	7200.13
	2				3025.09
	3				12.34
	4				10.81
	5				11.98
	6	71	100%	0 53	7200
	7				16.75
	8	118	100%	72	7200
	9	133	71.43%, [33]	69	7200
	10	60	34.33%, [24]	34	7200

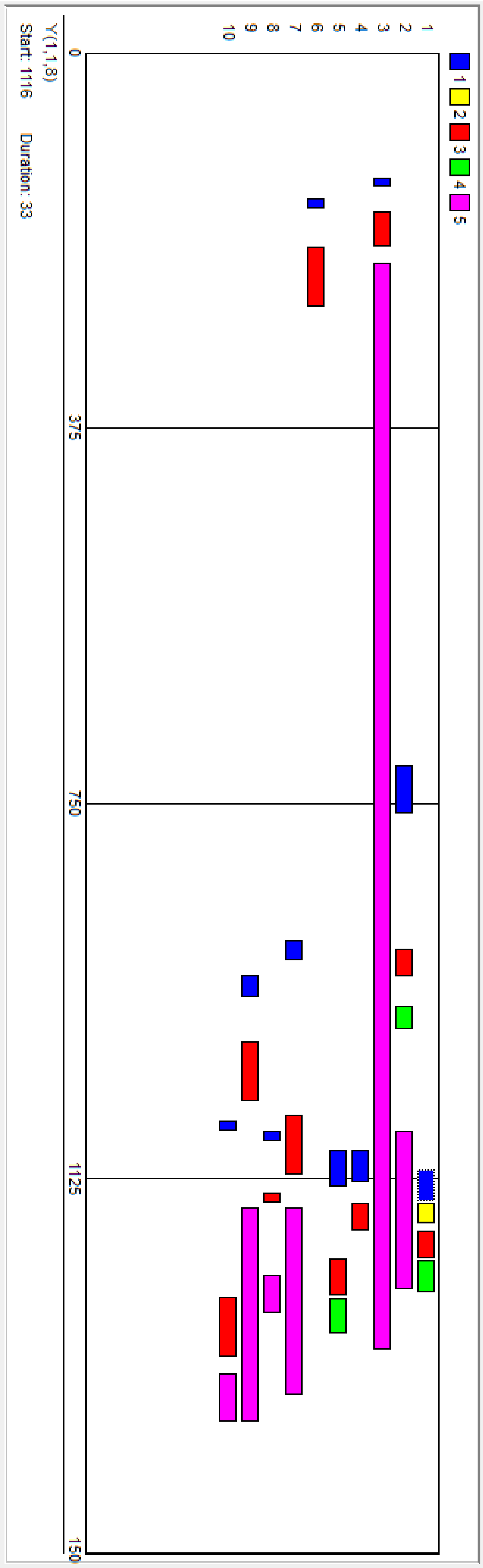

Figure 5.3: Optimal solution problem instance 10.1 part 1

	Waitingtime				
	1	2	3	4	5
1					
2				119	
3					
4					
5					
6				355	
7		74	118		
8					
9				14	
10		10			

Figure 5.4: Waiting time problem instance 10-1

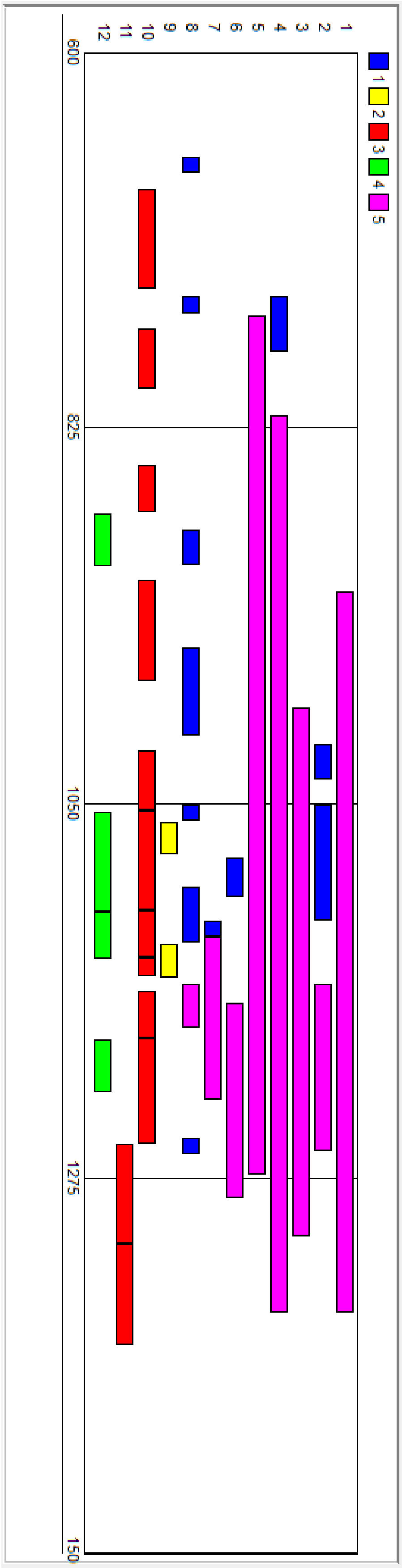

Figure 5.5: Optimal solution problem instance 12.1 part II

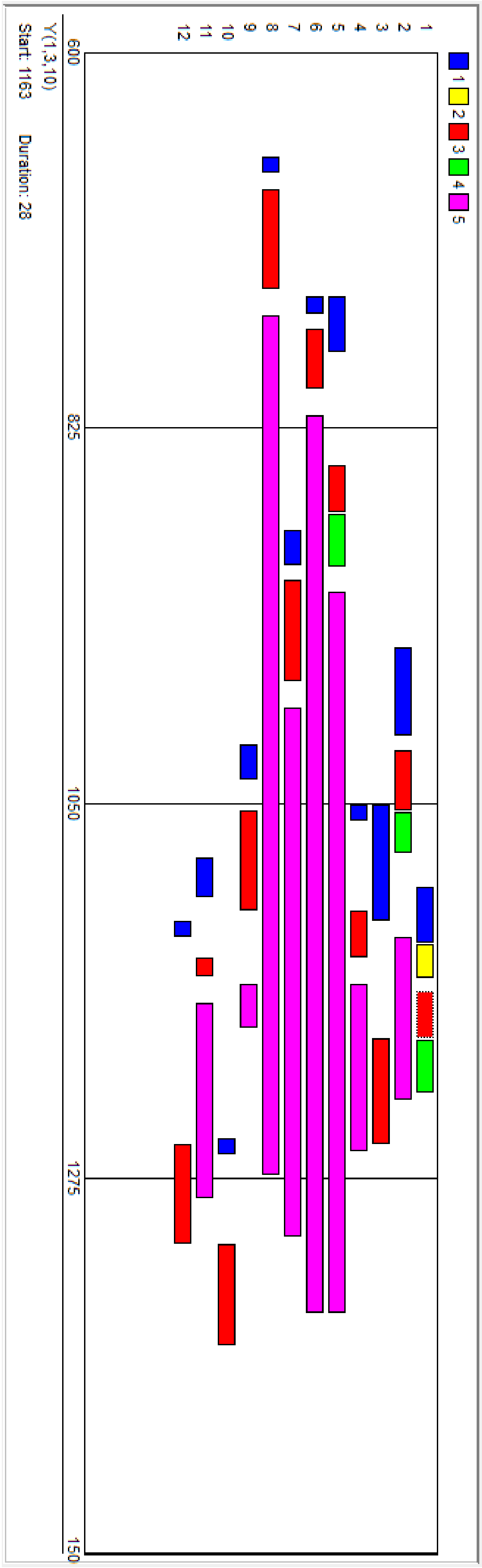

Figure 5.6: Plan problem instance 12.1 Part II

	Tardiness
1	18
2	
3	2
4	
5	
6	
7	
8	
9	
10	4
11	
12	34

Figure 5.7: Waiting time problem instance 10-1

5.3 Result conclusions

From Experiment 1 we conclude that the model can find optimal plans for problem instances up to 16 train units with different release and due dates within a few seconds. For only a few problem instances the model needs the full two hours. These problem instances are not solved to optimality. For only one of 160 problem instances no feasible solution exists. Problem instances containing more than 12 train units often experience a positive tardiness. This means that these scenario's cannot be used for NS. Planners in practice, choose to not process certain services in such a case but there is not method currently for making this decision in a clever way.

Experiment 2 plans jobs with a realistic set of first-line services. For this experiment the maximum number of train units for which plans can be found without tardiness is 16 train units, just like Experiment 1. However, in Experiment 2 some problem instances that include less than 16 train unit already a positive tardiness. The cause for the positive tardiness is that when train units require more first-line services a higher chance exists that other train units need to wait on them, before they can start processing their operation. Another reason for the tardiness of jobs is tight release and due dates.

The two experiments show that the model is not able to find feasible solutions for real life size problem instances of 28 train units. We saw that in practice also not all operations are executed. We can conclude, based on this model that this is indeed not possible. This model only takes into account the planning of the first-line services, it is even more difficult to find a plan when taking into account the other aspects.

We find that it is difficult to make a fair comparison between our results and the result by HIP and OPG. Both algorithms use a more comprehensive planning approach and thus have different results than our model. Also OPG approaches planning for first-line services very differently. When extending this model with for example the routing model that (Wolfhagen, 2017) developed a fairer comparison can be made. This is a recommendation for future research on which we elaborate in the next chapter.

6 | Conclusion & discussion

This chapter provides a brief overview of the outcomes of my study and what NS and model builders can learn from these. Section 6.1 draws conclusions from our research by answering the research question. Section 6.2 discusses the model and Section 6.3 presents the recommendations and suggestions for future research for NS.

6.1 Conclusion

We conduct this study to answer the following research question:

How can first-line services at NS service stations be planned, using an exact approach?

NS service station *Kleine Binckhorst* is our case SB for which we developed a model. We modeled the planning situation of SB *Kleine Binckhorst* as a flexible flow shop. We added extensions to the classical version of a flow shop problem to make it fit the NS situation, such as adding operations to jobs, and including the recirculation of jobs. By including these many features we create a more elaborate model than we see in literature. Most articles focus on two of three special features.

The model is an Integer Linear Programming (ILP) problem. We solve problem instances in AIMMS Optimization modeling software by using CPLEX 12.8. The model includes the most important restrictions and features that are present in reality and is extended by including travel times between machines. Problem instances containing up to 16 train units requiring a realistic set of first-line services result in feasible plans. The model can find solutions for problem instances larger than 16 train units, but these all have a positive tardiness, and are because of this, not useful for NS. Depending on the release and due dates of the train units it may be possible that problem instances exist for which feasible plans can be found. However, the results of this study show that given the release and due dates of the tested problem instances, problem instances containing more than 16 train units cannot be planned without trains experiencing delays.

Most solutions are generated within seconds. For supporting planners in creating a first-line services plan this is a very good feature. In practice up to 28 train units enter SB *Kleine Binckhorst* every day. This is more than our model can find a plan for. However, there is no insight in how many required first-line services of the 28 train units are planned. In our results, for all 16 train units realistic sets of required first-line services can be planned. Choosing which operations to plan and which not helps in planning more train units but do not fulfill the objective of planning all first line services. Based on the results of the model, 16 train units with a realistic set of train units is the capacity limit at SB *Kleine Binckhorst*.

6.2 Discussion

This section discusses the model, its limitations and whether it is worth the effort to continue developing the exact approach for solving the service station planning problem, based on the results of this study.

In the previous section we concluded that it is possible to plan first-line services of NS service stations using an exact approach. Finding solutions within a few seconds for up to 16 train units was above expectation because normally exact models can only be solved for small instances. We aggregated the nine first-line services into five operations. This helps in reducing the planning problem in size because now only five operations need to be planned. This assumption is also used in OPG. If train units are parked on tracks where multiple operations can be processed this happens consecutive. However, inspection A and B can be processed on the stabling tracks, in our model stage 1 & 5. In our model, we added the inspections to the arrival operation, but in practice this may also be executed just before the train unit leaves the SB. Furthermore, we divided the machines in groups and linked these to the operations. This way only a few machines need to be considered to plan an operation. This modeling simplification is as it is in practice. Only a few operations can be processed on certain tracks and these combinations are fixed and don't change. So in practice the planners do the same, they only consider a few tracks per operation.

Our model plans the first-line services as a round trip, for which the routing model needs yet to be integrated. This implies that the solutions found by the model are a bound for the service station planning problem. When extending the model, more constraints need to be taken into account. Therefore, the chances of finding a feasible plan decrease, while the computational requirement increase.

OPG, which is initially developed to solve the routing problem using an exact approach, can now also plan a few first-line services. Comparing our model with OPG, we see that OPG approaches the first-line services planning problem aspect very differently. OPG splits the first-line services planning problem into two parts, and solves each of the two parts sequentially. First, the problem of finding a route to, and a planning of the first-line services until the internal cleaning machine is solved Second, a planning of first-line services and a route back to the stabling tracks needs to be found. This model provides a different angle on how to solve the first-line services plan using an exact approach.

The problem instances that could be solved (to optimality), are smaller than realistic-size problem instances. We found that the model can find solutions for problem instances with $n \geq 18$ but that these problem instances all have a positive tardiness. Therefore, these problem instances are not useful for NS. In practice planners make choices in which first-line services to execute and which not, to preclude tardiness of train units. This is different than in our model. By processing only a set op required operations, the train units might be, for example, not as clean as NS would like. More important, it hinders the planning of the SB on which the train unit arrives the next day. All operations need to be executed within a certain time window, see Section 2. When an operation is not processed when it should, this operation needs to be prioritized on the next SB that that train unit arrives. This is an additional operation that that SB needs to plan. To prevent this from happening, we created the model such that all operations need to be processed. Choosing the operations should be done in a smart way to process as much operations as possible. However, before NS can improve their choices in which operations to plan and which not, a better understanding of the SB capacity should exists. Because, now the decisions are just based on the experience of the planner.

One of the most important motives for this research is the wish of NS to investigate the exact approach for solving the service station planning problem. Chapter 5 shows that our model can plan problem instances containing up to 16 train units with realistic sets of operations. The model takes into account the most relevant first-line services planning constraints and can be solved in a short amount of time. Now that we know how the first-line services can be planned, we can investigate whether it is worth to continue developing models for the service station planning problem using an exact approach. There are three aspects that we discuss.

First, the approach for solving the first-line services problem using an exact approach is very different compared to using a heuristic approach such as HIP. Also, exact approaches can find an optimum plan and by using a heuristic, this is never certain. Therefore, an exact approach has an advantage over the use of heuristics. However, heuristics are generally quicker in solving problem instances and can handle larger problem instances.

Second, before the model can be used in practice, it needs to be integrated with the other subproblems. These integrations create an even more complex problem because more aspects need to be considered, but also more constraints are added. This extends the required running time. Furthermore, extending the problem diminishes the chance on finding a feasible plan. This also holds for HIP. It is difficult to estimate at this point which of the two will perform better. Therefore, this should be tested by for example, integrating the model developed by (Wolfhagen, 2017) with this model.

Third, NS conducts research to solve the service station planning problem from all different angles. Machine learning is also one of the approaches that NS recently introduces. Machine learning needs solutions to learn from. These solutions might be results of a model based on a heuristic but if you have optimal solutions this helps the machine learning models to come up with even better solutions. For this, the results of this model may also be useful.

Finally, a combination between an exact approach and a heuristic might also be helpful in solving the first-line services planning problem. As we discussed earlier in this section, in practice planners decide which operations they can include in their planning and which they leave out. An heuristic can help in determining which first-line services to plan. For example, every required first-line service of a job might get a weight, based on how important or urgent it is to process that first-line service. Subsequently, a model using an exact approach can subsequently be used to solve this problem instance, and can maybe even find the optimal solution, depending on the allowed run time and problem instance.

Concluding, the development of the exact approach has multiple benefits and applications, and NS should continue this development. From this point in the development process of exact models, there are many steps that could be taken, or at least examined further, to investigate what the possibilities are in using an exact approach in solving the service station planning problem. In the next section we elaborate on what these possibilities are.

6.3 Recommendations and future research

In this subsection we present recommendations and suggestions for future research for NS. We describe extensions for the model to match reality closer and how to make the model applicable for other SBs.

First, our model includes the assumption that every job needs to visit all assigned machines, even though its operation on that machine might have a processing time of zero. This assumption created a more simple model and still is fairly realistic, because of the carousel layout at SB *Kleine Binckhorst*. However, all train units need to visit the machines that process operation 2. This is an operation that is hardly used in practice, as Table 5.1 shows. Every train unit needs to travel to and from this track (track 64, see Figure 4.2), increasing the time that train unit needs before it reaches its departure track. Also, in reality it is difficult to reach this track. We highly recommend to change the model such that, if a job has a processing time of zero for operation 2, it can skip this machine instead of how the model is currently set up, still has to visit this machine.

Second, a step further could be to extend the model by making it optional if the jobs need to visit the machines in a fixed sequence. By creating this option, the model can also be applied to shuffleboard lay-out SBs. At an SB with a shuffleboard layout, a fixed sequence of how the operations need to be processed might not be beneficial. For every operation that needs to be processed, a train unit movement is needed. Because all operations take place on different tracks, and train units cannot reach other maintenance tracks without driving back to a parking track a carousel layout is not necessarily needed. So, a more flexible order of processing train units can result in better plans for this type of lay-out.

Third, an extension to the model to come closer to solving the service station planning problem is to integrate the model developed in this study, with the model created by Wolfhagen (2017). Wolfhagen developed a model that focuses on the matching and parking of train units. Integrating this, adds a valuable dimension to the first-line services model.

Fourth, the model includes fixed travel times but the routing aspect of the model can be extended in multiple ways:

- Include different travel times for the different train types.
- Include like in reality, that only one train unit may travel over one trajectory at the time. This is because of safety regulations.
- Include travel restrictions based on whether there is enough room for the train units to travel directly for their current machine to their next machine. In this model we assume there is always enough room on the tracks, however,in reality this is not the case.

The fifth recommendation is based on the formulation of the model. Constraints may be reformulated using smarter ways of defining for example the Big-M parts of the equations. Including Big-M increases the solution space. Therefore, removing the Big-M might also help in solving the program quicker. However, we expect that this currently does not have any significant impact, because already most problem instance can be solved within seconds. When extending the model, this might become more important.

The sixth and last recommendation for extending the model is to include the limitations of the personnel planning. The limitation of one cleaning crew is already included, but there are a lot more examples in practice. For example, at stage 1 and 5, we have 8 tracks, and on all tracks operations may be executed at the same time. In reality, only three mechanics at most are available to process services. Here, a distinction needs be made between the arrival and departure operations, and the inspections since for the arrival and departure operations no mechanics are needed.

Finally, this model is a next step in the process of solving the NS service station planning problem using an exact approach. By providing these recommendations and suggestions, we hope that this thesis is an inspiration for others to continue this path.

A | Overview of travel times between machines

Table A.1 shows the travel times between the machines.

Table A.1: Travel times between machines in minutes

		From machine											
		1	2	3	4	5	6	7	8	9	10	11	12
To machine	1	0	7	7	7	8	8	8	8	11	14	14	15
	2	7	0	7	7	8	8	8	8	11	14	14	15
	3	7	7	0	7	8	8	8	8	11	14	14	15
	4	7	7	7	0	8	8	8	8	11	14	14	15
	5	8	8	8	8	0	10	10	10	13	4	4	5
	6	8	8	8	8	10	0	10	10	13	4	4	5
	7	8	8	8	8	10	10	0	10	12	4	4	5
	8	8	8	8	8	10	10	10	0	1	4	4	5
	9	11	11	11	11	13	13	9	0	0	8	8	9
	10	14	14	14	14	4	4	4	4	8	0	0	1
	11	14	14	14	14	4	4	4	4	8	0	0	1
	12	15	15	15	15	5	5	5	5	9	1	1	0

B | input Experiment 1

This appendix describes how we transformed the times of the problem instances of the test cases to come to useful input data for our model. The test cases are generated by Roel van der Broek and look like Table B.1:

Of this table we use the time and the train type of every train unit. We matched the arrival and departure times in such a way that train units of an equal train type visit the SB at least two hours. To translate the times to useful time for the model we use two equations in Excel shown by Equation B.1. Within the brackets, in place of 'TIMECELL' the cell of the time that we want to transform is placed. Equation B.1a shows the equation that we use when the time is before 00.00 hour. Equation B.1b is used when the time is after 00.00 hour. We need two equations because we do not only need to transform the time into seconds but also skew the time, because the day in the model starts at 08.00 hour (0 minutes) to 08.00 hour (1440 minutes) the next day.

$$= \text{HOUR}(TIMECELL) * 60 + \text{ROUNDUP(SECOND}(TIMECELL/60); 1) - 480 \tag{B.1a}$$

$$= \text{HOUR(TIMECELL)} * 60 + \text{ROUNDUP(SECOND(TIMECELL}/60); 1) + 960 \tag{B.1b}$$

We did this transformation for all times that form the release and due dates of all train units.

Table B.1: Example problem instance, first problem instance of test case n=10

Arrival					
time	Train unit number	train	position	type	track
02:43:57	2401	1000	0	SLT4	906a
19:52:12	2402	2000	0	SLT4	906a
10:04:42	2601	3000	0	SLT6	906a
02:16:54	2403	4000	0	SLT4	906a
02:16:54	2602	4000	1	SLT6	906a
10:25:52	9401	5000	0	VIRM4	906a
22:47:12	9402	6000	0	VIRM4	906a
01:58:27	8601	7000	0	VIRM6	906a
23:22:32	9403	8000	0	VIRM4	906a
23:22:32	9404	8000	1	VIRM4	906a
Departure					
time	train	position	type	track	
04:35:36	51000	0	SLT4	906a	
04:35:36	51000	1	SLT6	906a	
05:35:53	52000	0	SLT4	906a	
05:35:53	52000	1	SLT4	906a	
05:35:53	52000	2	SLT6	906a	
06:22:08	53000	0	VIRM4	906a	
23:06:12	54000	0	VIRM4	906a	
05:00:28	55000	0	VIRM6	906a	
06:47:30	56000	0	VIRM4	906a	
06:47:30	56000	1	VIRM4	906a	

C | input Experiment 2

This appendix shows the input for the first-line services and thus the operations for the jobs within the different test cases. Table C.1 shows how the number of train units that require a certain first-line services. Tables C.2, C.3, C.4, C.5 show the input for the jobs in the problem instances for in AIMMS

Table C.1: Overview of the total required first-line services per problem instance

	Operation:								
	1			**2**		**3**	**4**		**5**
Problem instance:	1	2	3	4	5	6	7	8	9
n=2	2	1	1	1	1	2	1	1	2
n=4	4	2	1	1	1	4	1	1	4
n=6	6	3	1	1	1	6	1	1	6
n=8	8	4	1	1	1	8	2	1	8
n=10	10	5	1	1	1	10	2	1	10
n=12	12	6	1	1	1	12	2	1	12
n=14	14	7	2	1	1	14	2	1	14
n=16	16	8	2	1	1	16	3	1	16
n=18	18	9	2	1	1	18	3	1	18

Table C.2

Test case: n=2										
	Operation:	1			2		3	4		5
	First-line service:	1	2	3	4	5	6	7	8	9
Problem instance:	1	1	1		1	1	1	1		1
	2	1	0	1			1		1	1

Test case: n=4										
	Operation	1			2		3	4		5
	First-line service:	1	2	3	4	5	6	7	8	9
Problem instance:	1	1	1		1	1	1	1		1
	2	1	0	1			1		1	1
	3	1	1				1			1
	4	1					1			1

Test case: n=6										
	Operation:	1			2		3	4		5
		1	2	3	4	5	6	7	8	9
Problem instance:	1	1	1		1	1	1	1		1
	2	1	0	1			1		1	1
	3	1	1				1			1
	4	1	0				1			1
	5	1	1				1			1
	6	1					1			1

Table C.3

Test case: n=8										
	Operation:	1			2		3	4		5
	First-line service:	1	2	3	4	5	6	7	8	9
Problem instance:	1	1	1		1	1	1	1		1
	2	1	0	1			1		1	1
	3	1	1				1			1
	4	1	0				1			1
	5	1	1				1	1		1
	6	1	0				1			1
	7	1	1				1			1
	8	1					1			1

Test case: n=10										
	Operation:	1			2		3	4		5
	First-line service:	1	2	3	4	5	6	7	8	9
Problem instance:	1	1	1		1	1	1	1		1
	2	1	0	1			1		1	1
	3	1	1				1			1
	4	1	0				1			1
	5	1	1				1	1		1
	6	1	0				1			1
	7	1	1				1			1
	8	1					1			1
	9	1	1				1			1
	10	1					1			1

Test case: n=12										
	Operation:	1			2		3	4		5
	First-line service:	1	2	3	4	5	6	7	8	9
Problem instance:	1	1	1		1	1	1	1		1
	2	1	0	1			1		1	1
	3	1	1				1			1
	4	1	0				1			1
	5	1	1				1	1		1
	6	1	0				1			1
	7	1	1				1			1
	8	1	0				1			1
	9	1	1				1			1
	10	1					1			1
	11	1	1				1			1
	12	1					1			1

Table C.4

Test case: n=14										
	Operation:	1			2		3	4		5
	First-line service	1	2	3	4	5	6	7	8	9
Problem instance:	1	1	1		1	1	1	1		1
	2	1	0	1			1		1	1
	3	1	1				1			1
	4	1	0				1			1
	5	1	1				1	1		1
	6	1	0				1			1
	7	1	1				1			1
	8	1	0				1			1
	9	1	1				1			1
	10	1					1			1
	11	1	1				1			1
	12	1					1			1
	13	1	1				1			1
	14	1					1			1
Test case: n=16										
	Operation:	1		2		3	4		5	
	First-line service:	1	2	3	4	5	6	7	8	9
Problem instance:	1	1	1		1	1	1	1	1	1
	2	1	0	1			1			1
	3	1	1				1			1
	4	1	0				1			1
	5	1	1				1	1		1
	6	1	0				1			1
	7	1	1				1			1
	8	1	0				1			1
	9	1	1				1	1		1
	10	1					1			1
	11	1	1				1			1
	12	1					1			1
	13	1	1				1			1
	14	1					1			1
	15	1	1				1			1
	16	1					1			1

Table C.5

Test case: n=18										
	Operation:	1			2		3	4		5
	First-line service:	1	2	3	4	5	6	7	8	9
Problem instance:	1	1	1		1	1	1	1	1	1
	2	1	0	1			1			1
	3	1	1				1			1
	4	1	0				1			1
	5	1	1				1	1		1
	6	1	0				1			1
	7	1	1				1			1
	8	1	0				1			1
	9	1	1				1	1		1
	10	1					1			1
	11	1	1				1			1
	12	1					1			1
	13	1	1				1			1
	14	1					1			1
	15	1					1			1
	16	1					1			1
	17	1					1			1
	18	1					1			1

D | Problem instance results Experiment 1 & Experiment 2

Tables D.2 and D.1 show the results of the problem instances of the test cases for which $n = 2, 4, 6, \&8$ of which a Tables are the summary off.

Table D.1: Results problem instances of test cases n=(2-8) of Experiment 2

		Obj. (min)	Job's max tardiness (min)	Time (s)
n=2	1			0,08
	2			0,08
	3			0,09
	4			0,08
	5			0,09
	6			0,08
	7			0,08
	8			0,09
	9			0,06
	10			0,08
n=4	1			0,24
	2			0,23
	3			0,20
	4			0,25
	5			0.22
	6			0,25
	7			0,22
	8			0,26
	9			0,17
	10			0,22
n=6	1			0,66
	2			0,36
	3			0.34
	4			0.28
	5			0.50
	6			0.25
	7			0.38
	8			0.27
	9			0.30
	10			0.22
n=8	1			0,52
	2			0,48
	3			0,48
	4			0,45
	5			1,03
	6			0,47
	7			1.30
	8			0.45
	9			0.50
	10			0.81

Table D.2: Results problem instances of test cases n=(2-8) of Experiment 1

		Obj. (min)	Job's max tardiness (min)	Time (s)
n=2	1			0.11
	2			0.05
	3			0.06
	4			0.09
	5			0.09
	6			0.19
	7			0.06
	8			0.08
	9			0.09
	10			0.08
n=4	1			0.44
	2			0.70
	3			0.36
	4			0.34
	5			0.31
	6			0.21
	7			0.63
	8			0.38
	9			0.28
	10			0.38
n=6	1			1.22
	2			1.09
	3			1.36
	4			1.66
	5			2.09
	6			1.03
	7			1.08
	8			1.58
	9			1.03
	10			1.25
n=8	1			4.53
	2			4.58
	3			3.44
	4			2.11
	5			8.94
	6			4.67
	7			4.38
	8			4.28
	9			4.20
	10			4.13

References

Ahonen, H. and de Alvarenga, A. G. (2017). Scheduling flexible flow shop with recirculation and machine sequence-dependent processing times: formulation and solution procedures. *The International Journal of Advanced Manufacturing Technology*, 89(1-4):765–777.

Al-Fawzan, M. A. and Haouari, M. (2005). A bi-objective model for robust resource-constrained project scheduling. *International Journal of production economics*, 96(2):175–187.

Albrecht, A. R. et al. (2009). *Integrating railway track maintenance and train timetables*. PhD thesis, University of South Australia.

association for Project Management (2017). Resource scheduling.

Bellenguez, O. and Néron, E. (2004). Lower bounds for the multi-skill project scheduling problem with hierarchical levels of skills. In *International Conference on the Practice and Theory of Automated Timetabling*, pages 229–243. Springer.

Blasum, U., Bussieck, M. R., Hochstättler, W., Moll, C., Scheel, H.-H., and Winter, T. (1999). Scheduling trams in the morning. *Mathematical Methods of Operations Research*, 49(1):137–148.

Bojovic, N. and Milenkovic, M. (2010). Train driver rostering optimization. In *Intelligent Systems and Informatics (SISY), 2010 8th International Symposium on*, pages 501–504. IEEE.

Brčić, M., Kalpić, D., and Fertalj, K. (2012). Resource constrained project scheduling under uncertainty: a survey. In *23rd Central European Conference on Information and Intelligent Systems*.

Cadarso, L. and Marín, Á. (2010). Robust routing of rapid transit rolling stock. *Public Transport*, 2(1):51–68.

Camm, J. D., Raturi, A. S., and Tsubakitani, S. (1990). Cutting big m down to size. *Interfaces*, 20(5):61–66.

Cheng, Y.-H. and Tsao, H.-L. (2010). Rolling stock maintenance strategy selection, spares parts' estimation, and replacements' interval calculation. *International Journal of Production Economics*, 128(1):404–412.

Cordeau, J.-F., Laporte, G., Pasin, F., and Ropke, S. (2010). Scheduling technicians and tasks in a telecommunications company. *Journal of Scheduling*, 13(4):393–409.

Corman, F., Kraijema, S., Godjevac, M., and Lodewijks, G. (2017). Optimizing preventive maintenance policy: A data-driven application for a light rail braking system. *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of risk and reliability*, 231(5):534–545.

Cornelsen, S. and Di Stefano, G. (2007). Track assignment. *Journal of Discrete Algorithms*, 5(2):250–261.

den Ouden, J. (2018). Generating robust schedules for train maintenance staff. Master's thesis.

Di Stefano, G. and Koči, M. L. (2004). A graph theoretical approach to the shunting problem. *Electronic Notes in Theoretical Computer Science*, 92:16–33.

Dutot, P.-F., Laugier, A., and Bustos, A.-M. (2006). Technicians and interventions scheduling for telecommunications. *France Telecom R&D*.

Fioole, P.-J., Kroon, L., Maróti, G., and Schrijver, A. (2006). A rolling stock circulation model for combining and splitting of passenger trains. *European Journal of Operational Research*, 174(2):1281–1297.

Fırat, M. and Hurkens, C. (2012). An improved mip-based approach for a multi-skill workforce scheduling problem. *Journal of Scheduling*, 15(3):363–380.

Freling, R., Lentink, R. M., Kroon, L. G., and Huisman, D. (2005). Shunting of passenger train units in a railway station. *Transportation Science*, 39(2):261–272.

Gallo, G. and Miele, F. D. (2001). Dispatching buses in parking depots. *Transportation Science*, 35(3):322–330.

Gestrelius, S., Aronsson, M., Joborn, M., and Bohlin, M. (2017). Towards a comprehensive model for track allocation and roll-time scheduling at marshalling yards. *Journal of Rail Transport Planning & Management*.

Giacco, G. L., Carillo, D., D'Ariano, A., Pacciarelli, D., and Marín, Á. G. (2014). Short-term rail rolling stock rostering and maintenance scheduling. *Transportation Research Procedia*, 3:651–659.

Gupta, J. N. (1988). Two-stage, hybrid flowshop scheduling problem. *Journal of the Operational Research Society*, 39(4):359–364.

Haijema, R., Duin, C., and van Dijk, N. (2006). *Train Shunting: A Practical Heuristic Inspired by Dynamic Programming*, pages 437–475. John Wiley & Sons, Inc.

Hans, E. W., Van Houdenhoven, M., and Hulshof, P. J. (2012). A framework for healthcare planning and control. In *Handbook of healthcare system scheduling*, pages 303–320. Springer.

Haouari, M., Hidri, L., and Gharbi, A. (2006). Optimal scheduling of a two-stage hybrid flow shop. *Mathematical Methods of Operations Research*, 64(1):107–124.

Jafarzadeh, H., Moradinasab, N., and Gerami, A. (2017). Solving no-wait two-stage flexible flow shop scheduling problem with unrelated parallel machines and rework time by the adjusted discrete multi objective invasive weed optimization and fuzzy dominance approach. *Journal of Industrial Engineering and Management*, 10(5):887.

Jain, V. and Grossmann, I. E. (2001). Algorithms for hybrid milp/cp models for a class of optimization problems. *INFORMS Journal on computing*, 13(4):258–276.

Jędrzejowicz, P. and Ratajczak-Ropel, E. (2014). Reinforcement learning strategies for a-team solving the resource-constrained project scheduling problem. *Neurocomputing*, 146:301–307.

Kolisch, R. and Drexl, A. (1996). Adaptive search for solving hard project scheduling problems. *Naval Research Logistics (NRL)*, 43(1):23–40.

Kovacs, A. A., Parragh, S. N., Doerner, K. F., and Hartl, R. F. (2012). Adaptive large neighborhood search for service technician routing and scheduling problems. *Journal of scheduling*, 15(5):579–600.

Kroon, L. G., Lentink, R. M., and Schrijver, A. (2008). Shunting of passenger train units: an integrated approach. *Transportation Science*, 42(4):436–449.

Lentink, R. M., Fioole, P.-J., Kroon, L. G., and van't Woudt, C. (2006). Applying operations research techniques to planning of train shunting. *Planning in Intelligent Systems: Aspects, Motivations, and Methods*, pages 415–436.

Michalewicz, Z. and Fogel, D. B. (2004). Why are some problems difficult to solve? In *How to Solve It: Modern Heuristics*, pages 11–30. Springer.

Nederlandse Spoorwegen (2016). Vervoerplan 2017. Report, Nederlandse Spoorwegen.

Nederlandse Spoorwegen (2017a). Leve de koning!

Nederlandse Spoorwegen (2017b). Oorlog en fusie.

NS (2015). internal document. Technical report, Group, NS.

NT Operations (2015a). Werkbeschrijving a-controle. Report, NedTrain.

NT Operations (2015b). Werkbeschrijving b-controle. Report, NedTrain.

Peeters, M. and Kroon, L. (2008). Circulation of railway rolling stock: a branch-and-price approach. *Computers & operations research*, 35(2):538–556.

Peng, F. and Ouyang, Y. (2014). Optimal clustering of railroad track maintenance jobs. *Computer-Aided Civil and Infrastructure Engineering*, 29(4):235–247.

Penicka, M., Strupchanska, A. K., and Bjørner, D. (2003). Train maintenance routing. In *FORMS'2003: Symposium on Formal Methods for Railway Operation and Control Systems*.

Ruiz, R., Şerifoğlu, F. S., and Urlings, T. (2008). Modeling realistic hybrid flexible flowshop scheduling problems. *Computers & Operations Research*, 35(4):1151–1175.

Ruiz, R. and Vázquez-Rodríguez, J. A. (2010). The hybrid flow shop scheduling problem. *European Journal of Operational Research*, 205(1):1–18.

Samà, M., D'Ariano, A., Pacciarelli, D., and Corman, F. (2016). Lower and upper bound algorithms for the real-time train scheduling and routing problem in a railway network. *IFAC-PapersOnLine*, 49(3):215–220.

Schutt, A., Feydy, T., Stuckey, P. J., and Wallace, M. G. (2011). Explaining the cumulative propagator. *Constraints*, 16(3):250–282.

Schutten, M. (1996). Shop floor scheduling with setup times. *these de l'Uiversite of Twente*.

Sriskandarajah, C., Jardine, A., and Chan, C. (1998). Maintenance scheduling of rolling stock using a genetic algorithm. *Journal of the Operational Research Society*, 49(11):1130–1145.

Trautmann, N. and Baumann, P. (2009). Resource-allocation capabilities of commercial project management software: An experimental analysis. In *Computers & Industrial Engineering, 2009. CIE 2009. International Conference on*, pages 1143–1148. IEEE.

van Dommelen, M. T. (2015). Scheduling train service activities at service sites to determine the capacity. Master's thesis, Vrije Universiteit Amsterdam.

Wagenaar, J., Kroon, L., and Fragkos, I. (2017). Rolling stock rescheduling in passenger railway transportation using dead-heading trips and adjusted passenger demand. *Transportation Research Part B: Methodological*, 101:140–161.

Wang, H. (2005). Flexible flow shop scheduling: optimum, heuristics and artificial intelligence solutions. *Expert Systems*, 22(2):78–85.

Winter, T. and Zimmermann, U. T. (2000). Real-time dispatch of trams in storage yards. *Annals of Operations Research*, 96(1):287–315.

Wolfhagen, F. (2017). The Train Unit Shunting Problem with Reallocation. Master's thesis, Erasmus University Rotterdam.

Yang, L., Li, K., and Gao, Z. (2009). Train timetable problem on a single-line railway with fuzzy passenger demand. *IEEE Transactions on fuzzy systems*, 17(3):617–629.