

MASTER'S THESIS ECONOMICS & INFORMATICS

---

# Train shunt planning using genetic algorithms

---

*Author:*

Dirk JEKKERS  
292493

*Supervisor:*

Prof.Dr.Ir. Uzay KAYMAK

*Second supervisor:*

Prof.Dr. Leo G. KROON

Master program: Computational Economics  
Erasmus School of Economics, Erasmus University Rotterdam  
April 9, 2009



## **Abstract**

Because demand for train transportation is very high during rush hours, train operators need a lot of rolling stock to supply this demand. Outside the rush hours, especially during the night, this results in a surplus of rolling stock which need to be parked at a shunt yard. Because of limit track capacity and the diversity in train types, scheduling this process is a hard and time consuming task. We have developed a tool that is able to generate good shunt schedules to support local shunt planners. With the use of a genetic algorithm, we were able to extend the basic problem formulation with flexible shunt times for each shunt activity in the schedule. Our model is successfully tested on a series of theoretical cases and two real-world cases from Netherlands Railways.



# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Relevance and Objective . . . . .	2
1.2	Research Questions . . . . .	2
1.3	Methodology . . . . .	2
1.4	Thesis outline . . . . .	2
<b>2</b>	<b>The Train Unit Shunting Problem</b>	<b>5</b>
2.1	The shunting problem . . . . .	5
2.2	Decision variables, restrictions and objectives . . . . .	8
2.3	Examples . . . . .	9
2.4	Related work . . . . .	11
2.4.1	General shunting problems . . . . .	12
2.4.2	The shunting problem at Netherlands Railways . . . . .	13
<b>3</b>	<b>The MIP Solution</b>	<b>17</b>
3.1	The Basic MIP model . . . . .	18
3.2	Evaluation of the MIP model . . . . .	20
3.3	Requirements for a new approach . . . . .	21
<b>4</b>	<b>A Metaheuristic Solution</b>	<b>23</b>
4.1	TUSP formulation . . . . .	24
4.2	Genetic Algorithm . . . . .	24
4.2.1	Chromosome representation . . . . .	26
4.2.2	Initial population . . . . .	29
4.2.3	Fitness evaluation . . . . .	29
4.2.4	Selection . . . . .	31
4.2.5	Reproduction . . . . .	31
4.3	Alternative Approach . . . . .	35
4.3.1	A different point of view . . . . .	37
4.3.2	Rule based decision . . . . .	38
4.3.3	Adjustments for the GA operators . . . . .	39
4.4	Simulation Model . . . . .	39
4.4.1	Components . . . . .	40
4.4.2	Performance statistic . . . . .	40
4.4.3	Events . . . . .	41

4.4.4	Verification of the simulation model . . . . .	43
<b>5</b>	<b>Experiments: Theoretical cases</b>	<b>45</b>
5.1	The test cases . . . . .	45
5.2	Experiments . . . . .	48
5.3	Results . . . . .	49
5.4	Conclusion . . . . .	56
<b>6</b>	<b>Experiments: Real-world cases</b>	<b>59</b>
6.1	Hoofddorp . . . . .	59
6.1.1	Experiments . . . . .	61
6.1.2	Results . . . . .	63
6.2	Rotterdam . . . . .	65
6.2.1	Experiments . . . . .	66
6.2.2	Results . . . . .	67
6.3	Applicability of the solution approach in real-world decision support . . . .	74
<b>7</b>	<b>Conclusion</b>	<b>75</b>
<b>A</b>	<b>Crossover Experiments</b>	<b>80</b>
<b>B</b>	<b>Timetables</b>	<b>82</b>

# Chapter 1

## Introduction

The need to park trains for a period of time occurs when demand for transport is not equally distributed over time. This happens every day for passenger train services because demand is huge during the rush hours and almost zero during nighttime. To supply the demand during the rush hours, a lot of resources (train units) are needed. Outside the rush hours, this results in a surplus of rolling stock. An efficient railway operator will park this idle rolling stock at a shunt yard. These shunt yards are mostly located next to a train station at major junctions, and have a limited capacity to store trains. For an efficient realization of the entire train schedule of the operator, it is necessary that a robust shunting schedule exists at each local shunt yard. Especially during the night when most of the operator's rolling stock is parked, it is important that the trains are parked in such a way that at the start of the next day, trains can easily be routed to their departure platform.

Until now, shunting schedules have been made mostly manually at several local planning offices and require a lot of labor and experience. Therefore operators are trying to find a way to compute shunting schedules automatically. The shunting schedule is dictated by the timetable, which describes the planned arriving and departure times of trains at a station. During the day most trains will continue servicing passengers after a stop at the station, but trains that are taken out of service have to be parked at a shunt track. When different trains are parked at the same shunt track, the order is of importance so that no train is blocking the arrival or departure of another train. Other aspects of the problem are the physical constraints of the infrastructure, necessary cleaning and maintenance processes, preferences for parking some train types at dedicated tracks, and providing a certain flexibility and robustness with regard to disturbances.

In this study we will focus on the shunting problem of Netherlands Railways (in Dutch: Nederlandse Spoorwegen, NS), the largest Dutch railway operator. Each day over 1,1 million passengers use the 4800 train services provided by NS [26]. The railroad network contains over 2700 km of track and with 380 stations it covers most of the Netherlands. At 30 shunt yards across the Netherlands, around 750 train units have to be parked every night. About 130 planners are currently working on the day-to-day planning of shunting operations.

## 1.1 Relevance and Objective

As part of the development of a new decision support system for local planners, NS is looking for a way to support planners in creating good shunt plans. Currently shunt plans are entirely made by hand. Because this is a time consuming process the individual shunt plans for each day are made far in advance. Because of changes in the timetable, rolling stock or track maintenance, each plan usually has to be modified a couple of times before the actual planned day starts. The aim of NS is start this scheduling process only a few days before the actual implementation, to reduce redundant work on schedules that have to be changed anyway. By providing planners with an intelligent support system that can generate a shunt schedule automatically, this process could in theory be postponed until the start of the planned period. The aim of this research is to develop a tool for automated shunt planning. This tool should be able to build feasible and good shunt plans that planners only need to examine and make minor adjustments if necessary.

## 1.2 Research Questions

To reach the objective of this research we will first have to examine the problem in detail and determine how we can model the shunting process. Furthermore we have to find a suitable algorithm/technique that can search for a good shunt plan. Our research questions are:

- How can we model the shunting problem such that we can evaluate different shunt plans?
- How can we efficiently find good and feasible shunt plans

## 1.3 Methodology

To answer the first research question, we look into the present literature on the shunting subject. We evaluate the approach that is currently under development at NS and use the comments of the developers as well as the shunt planners to formulate our own view on the shunting problem. Based on these findings we present a new approach. With experiments on different theoretical cases we try to answer the second research question. By evaluating the results of two different approaches with various settings of the implemented search technique we determine the best approach for this problem. Finally our new suggested approach is tested on two real-world cases and by examining the results we conclude whether our approach can generate good and feasible shunt plans.

## 1.4 Thesis outline

In the next chapter we start with an in-depth overview of the shunting problem and present related work in this field. Chapter 3 describes the approach currently in development at NS. In chapter 4 our new approach is introduced and the different methods and algorithms are explained. Chapter 5 presents the experiments and results for four theoretical cases.



In chapter 6 two real-world cases are described with experiments and results. Finally, we present the conclusions and give some ideas for further research in chapter 7.



## Chapter 2

# The Train Unit Shunting Problem

The job of a shunt planner is to plan a robust shunt schedule for a local shunt yard. For each arriving train at a station that will not continue its service for a period of time, he has to assign a shunt track on which the train will be parked. For each departing train at a station for which no train is in service, he has to assign a parked train at the shunt yard to start this service. The planner has to determine the route from the station to the shunt yard and back and choose the time in which this move has to be performed. The time of the shunt activity is primarily chosen such that there is track capacity available (i.e. there is a free route), but it could also help to reduce the workload of the shunting crew. For example, it could be useful to let trains wait on the platform until the next train arrives in order to shunt both train units together to the shunt yard. Shunting multiple units for different trains from the shunt yard to the platform at the same time could also reduce the workload for the shunting crew. For realising a better unit order on the shunt tracks or for cleaning purposes he can also move trains from one shunt track to another. Although this seems a quite trivial problem, planners have to take a lot of restrictions into consideration. For example, track capacity, train compositions and the parking order are all concerns that have to be taken into account. In this chapter we give a detailed description of the Train Unit Shunting Problem (TUSP). First the different aspects of the problem are explained, then we will discuss the decisions a planner has to make and finally we show a couple of examples to illustrate the problem.

### 2.1 The shunting problem

The shunting problem consists of a four different elements; train units, shunt location, timetable and the shunting crew.

**Train units** Most passenger trains operated by NS consist of one or more train units. A train unit is a rolling stock unit that can move bi-directionally without a locomotive. An operator often has different types of train units for different types of services (e.g. Inter-city trains, local trains). Train units of the same type differ from each other by the number of carriages per train unit, which we classify as different subtypes (see figure 2.1). Only train units of the same type can be combined to form longer train compositions. A train

composition can change through the day; in the rush hours for example, an extra unit can be added to handle the increase in demand. The order of units in a train composition is of importance; units of the same subtype can be swapped but units of different subtypes (i.e. different numbers of carriages) have to depart in the order that has been scheduled.



Figure 2.1: Two train units of the same type: ICM, but different subtypes: ICM3 and ICM4.

**Shunt location** With shunt locations we mean the stations with adjacent shunt yards and the surrounding railroad network. Each shunt location has its own characteristics, but in general they all have the following features; there is a station with different platforms at which trains arrive and depart. Most stations in the Netherlands can be approached from both sides. We call one side the “A-side” and the other the “B-side”. In general, the A-side is the side closest to Amsterdam. Next we have the shunt yard with different shunt tracks. The shunt tracks can either be at the A-side or the B-side of the station, some locations have shunt tracks at both sides. A shunt track can be divided into two categories; the “LIFO track” and the “free track”. A LIFO track can only be approached from one direction, which means that the last train that enters the shunt track is the first one that has to leave, hence the name LIFO (last in, first out). The free track can be approached from both sides. LIFO tracks can either be open at the A-side or the B-side. A-side open means that the side closest to the A side of the station is open, B-side open says that the side closest to the B side of the station is open. See figure 2.2 for a schematic drawing of a shunt location.

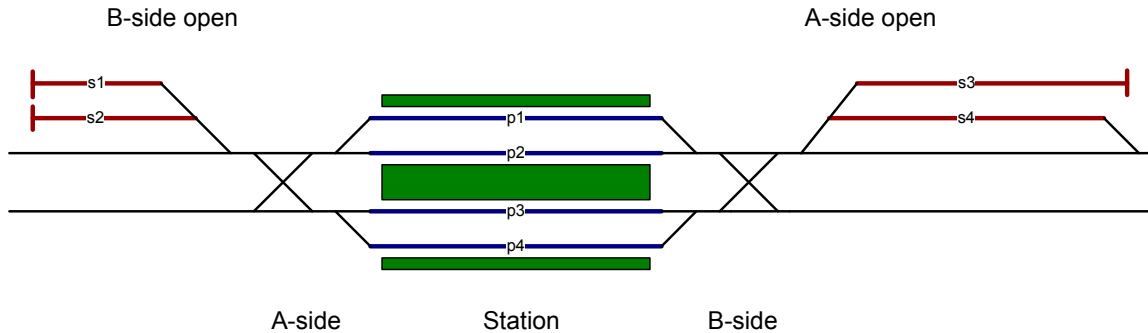


Figure 2.2: A typical shunt location with 4 platform tracks(p1-4), 3 LIFO shunt tracks(s1-3) and 1 free shunt track(s4)

**Timetable** The timetable of a shunt location describes at what time trains enter and leave the station. It describes the exact composition of each train and from which side it enters or leaves the station. In table 2.1 we show a timetable. In this example, a train

arrives with 3 units at platform 1, 10 minutes later this same composition leaves the station also at platform 1. In this case no shunt activity is needed. The second train enters the station at 10:30 with two units, 10 minutes later a train departs from the same platform with only one unit. In this case, we have to shunt the B unit to a shunt track. At 16:00 a train enters the station with one unit, 10 minutes later a train departs with two units at the same platform. This time we need an extra train unit that needs to be shunted from the shunt yard to the platform before the departure of the train. In the evening at 22:00 a train arrives at platform 1, half an hour later, without any departures at platform 1, a second train arrives at this platform. In this case the entire train from 22:00 will have to be shunted to a shunt track before the arrival of the second train.

Table 2.1: Timetable of a shunt location

Composition	Time	Arrive \ Depart	Side	Platform
A-A-B	10:00	Arrive	A	1
A-A-B	10:10	Depart	B	1
A-B	10:30	Arrive	A	2
A	10:40	Depart	A	2
B	16:00	Arrive	B	1
B-B	16:10	Depart	B	1
A-A	22:00	Arrive	A	1
B-A	22:30	Arrive	A	1

**Shunting crew** In general, shunt activities at the different locations are carried out by local shunting crew. When there is a limited amount of shunting crew available, the number of shunt moves performed at the same time is also limited. Moreover, when a train has been parked, the driver must walk to the track where the next shunt move has to be performed, so shunt moves following close after each other are also limited.

Other complications include cleaning services. Internal cleaning of train units has to be done every day and is often performed after the afternoon rush hour and before the start-up process in the next morning. The cleaning is preferably done at the end of a passenger line, just after the arrival of the train at a platform, and before it is shunted to the shunt yard. However this is not always possible due to new train arrivals. In that case, the train has to be parked (temporarily) at a shunt track with cleaning facilities. For the external cleaning, train units will have to be shunted to a track with a so-called train wash. This should be done once every 48 hours.

### List of terms

**Timetable:** a list of arrival and departure times for each train composition with the accompanied platform at the involved station.

**Train unit:** rolling stock that can move bi-directionally without a locomotive, capable of coupling with other units of the same type.

**Train composition:** train consisting of multiple train units, together forming a train composition. When the composition has units of different subtypes, the order of the units in the composition is of importance.

**Through train service:** A train that continues its service after a short stop at a station, or a train passing by without a stop.

**Shunt track:** a track where trains can be parked over a period of time.

- LIFO track, a shunt track that can only be approached from the A or B side
- Free track, a shunt track that can be accessed from both sides.

**Shunt yard:** the set of shunt tracks next to or nearby a station and the routes to each track.

**Railway station:** a place where passengers can get in or out of a train from a platform.

**Platform track:** a track next to a platform at the station.

**Shunt move:** moving a train unit from one track to another.

## 2.2 Decision variables, restrictions and objectives

Given the problem we described in section 2.1 a shunt planner has a couple of decisions he can make when building a shunt schedule:

- Assign arriving train units to a shunt track.
- Assign train units at the shunt yard to departing trains.
- Determine routes between shunt tracks and platforms.
- Select the start time of each shunt activity.
- Shunt parked train units between shunt tracks.

The planner is not entirely free when making these decisions. First of all he has to follow the timetable, train compositions should arrive and depart at the station in the given order at the scheduled time. Railway infrastructure is making train movements very restricted. A train can not pass another train on the same track, and for safety reasons, trains are not allowed to follow other trains too close on the same track. This also counts for passing of switches. These are the restrictions the planners face when making a schedule:

- Each shunt track can only store a limited amount of train units at the same time, limited by the length of the track
- A shunt move can only be performed when there is a free route to the other track.
- The composition of the departing train has to be in the exact order as planned in the timetable.

- Trains can not use the same track too close after each other.

A schedule should satisfy all these restrictions, but a planner also wants to make a “good” schedule. A schedule is considered to be good when the shunting effort is reduced, train compositions are kept together as much as possible, location preferences are taken into account and the number of shunting crew that is needed to perform the actual shunting process is low. The objectives of a good shunt schedule are:

- Arriving train compositions should be parked on the same shunt track.
- Departing train compositions should be retrieved from the same shunt track.
- Trains with a preferred shunt track should be parked at this track.
- Keep the number of shunt moves as small as possible.
- Create feasible schedules for the available shunting crew.

The above objectives are of a conflicting nature, meaning that by improving one objective an other objective can get worse. Different shunt locations have different priorities, so it is up to the planners to decide which objective is more important than the other. The optimal shunt schedule, given a station with its associated shunt yard and timetable, should apply these preferences as much as possible without exceeding the constraints. In the next section, to illustrate the complexity of the problem, we give a couple of examples that can occur when creating a schedule.

## 2.3 Examples

In this section we will give three small examples of typical problems that arise when making a shunt schedule. For each of the following examples we will use the same station and shunt yard, illustrated in figure 2.3. The station has two platforms, p1 and p2. The shunt yard has two LIFO shunt tracks, s1 and s2.

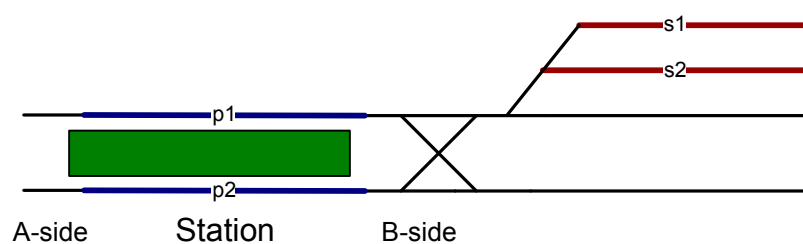


Figure 2.3: Track layout examples

**Example 1** Let us first consider a simple problem where three trains have to be parked at the end of the day, and have to leave the next morning. Table 2.2 shows the timetable of these three trains

Table 2.2: Timetable, example 1

Arriving Trains					Departing Trains				
Nr	Platform	Side	Time	Units	Nr	Platform	Side	Time	Units
1001	P1	A	mo 22:00	A	2001	P1	B	tu 07:00	A
1002	P2	B	mo 22:20	B	2002	P2	A	tu 07:30	C
1003	P1	A	mo 22:40	C	2003	P1	B	tu 07:40	B

Let us now park train 1001 and 1002 at track s2 and 1003 at track s1 (see figure 2.4a). The next day, train unit A has to leave first, however the train unit of type B is blocking the path to the platform because he was parked later at the shunt track. This means that train unit B has to be shunted to another track before train unit A can leave the shunt track. A better solution would be to park train 1001 to track s1 and 1002 and 1003 to track s2 (see figure 2.4b), now no train is blocking other trains, since a train unit C is needed before unit B. The situation where one unit is blocking the other on a shunt track is often referred to as 'crossing'.

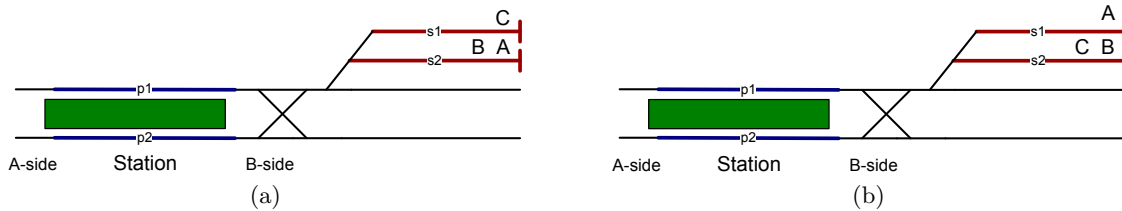


Figure 2.4: Example 1

**Example 2** In the next example we have a train composition with 2 different subtypes that has to be parked at the end of the day. The next day a departing train of the same composition is leaving in the opposite direction (i.e. the train enters and leaves the station from the same side). See table 2.3 for the timetable.

Table 2.3: Timetable, example 2

Arriving Trains					Departing Trains				
Nr	Platform	Side	Time	Units	Nr	Platform	Side	Time	Units
1001	P1	A	mo 22:00	AB	2001	P2	A	tu 07:00	AB

We could park the train composition without uncoupling the two train units at track s2 (see figure 2.5a), however, when we use this composition the next day, it is in the wrong order because it is going in the opposite direction. This would mean that we have to park unit B temporally at shunt track s1 so that we can first shunt unit A to the platform. A better solution would be to uncouple the train at the platform after its arrival, then park unit A at track s2 and unit B at track s1 (see figure 2.5b). The next day we would first shunt unit A to platform P2 and then couple unit B to A at the platform. The question here is where do we want to reduce the shunting effort, just after the arrival of train units or



just before the departure. In general planners choose to plan more shunting activities after the arrivals (in the evening) to save shunt moves before the departure (in the morning).

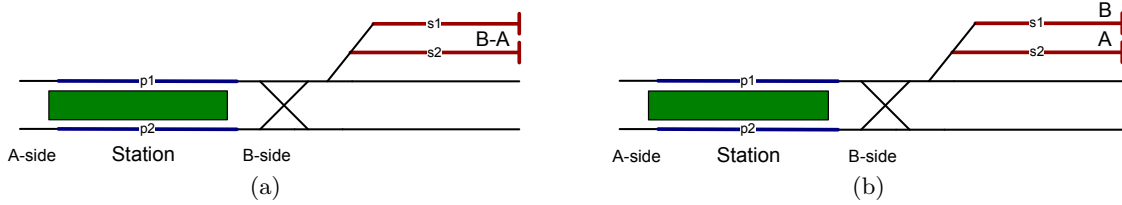


Figure 2.5: Example 2

**Example 3** In the last example, two trains arrive separately at the end of the day, but the next day they will depart as one composition. The first unit that arrives will also be in the front of the composition (see table 2.4). If we would park unit B immediately at the shunt yard, we would have to make an extra shunt move to put unit B in front of unit A. It would be better to let unit B wait at the platform until unit A has been parked at a shunt track (see figure 2.6). However, this is only possible when there are no other trains using platform 1 between 22:00 and 22:10.

Table 2.4: Timetable, example 3

Arriving Trains					Departing Trains				
Nr	Platform	Side	Time	Units	Nr	Platform	Side	Time	Units
1001	P1	B	mo 22:00	B	2001	P2	A	tu 07:00	BA
1002	P2	A	mo 22:10	A					

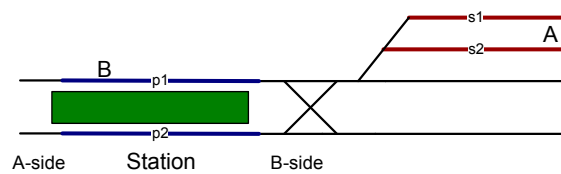


Figure 2.6: Example 3

## 2.4 Related work

In this section a review of the literature concerning (train) shunting is given. In the first part of this section the reviewed articles describe shunting problems that somewhat differ from the TUSP. These articles give an idea of the different approaches that have been used to solve shunting problems. In the second part of this chapter, the reviewed articles describe different approaches to the shunting problem at Netherlands Railways (NS). These approaches were derived from a project called Rintel in which NS invited a number of

researchers to look from their perspective at the shunting problem. One of these approaches was chosen for further development and is discussed in chapter 3.

### 2.4.1 General shunting problems

In the literature, Blasum et al. [8] were one of the firsts to discuss a typical shunting problem. In their research the problem of dispatching trams in a depot with only LIFO tracks is described. Incoming trams at the end of the day have to be assigned to a shunt track at the depot and parked trams have to be assigned to starting round trips the next day. The authors only focus on a sub problem; the assignment of stored trams to the scheduled round trips, minimizing the number of shunt moves. This is solved with the use of dynamic programming. In Winter and Zimmermann [34] the assignment of arriving trams is also discussed. First a method for finding the optimal solution is presented. However because actual arrival times can differ substantially from the scheduled times, they have developed a method that can make real-time decisions, using information of the optimal schedule and incorporating the new real-time situation of trams arriving in a different order. Gallo and Miele [15] have used the approach of Winter and Zimmermann to solve the dispatching problem for buses in a parking depot. They have extended the model by introducing vehicles of different lengths.

Tomii et al [32] combine a probabilistic local search method with a PERT (Program Evaluation and Review Technique) network to generate feasible shunting schedules. Their problem is of a different nature than the problem we described in section 1. First of all they only consider complete trains instead of train units. These trains cannot change their composition, and only one train is allowed to park at a shunt track at the same time. In their approach they first find an initial solution by using heuristics that assign trains to a shunt track with an accompanied shunting time (i.e. time when shunt move is performed). The solution might not be acceptable since this heuristic assigns shunt tracks depending on the arrival and departure track of the train, not taking any conflicts that can occur on those tracks into account. They represent the shunt schedule as a PERT network (directed graph) where the nodes correspond to an event of arrival or departure of a train, and the arcs express the time base order of execution between nodes. The weight of an arc means the minimum time necessary between the execution of the two events (i.e. the two nodes connected by this arc) and the direction tells which event should be executed first. In other words, an event can only be executed when its predecessor in the network is executed. The network is constructed with the initial solution that tells the order of execution. The network is tested on feasibility, and starting with the first unsatisfied constraint, a local search algorithm is used to find a new path that does satisfy the constraints. Tomii and Zhou [31] have extended this last model by adding tasks that have to be performed at a shunt track (e.g. cleaning), and which can only be performed on some of the tracks. These task are also represented as node in the network. In addition they include the scheduling of workforces to each shunt move and task that has to be completed, resulting in a lot of new constraints about assigning workers to tasks. Once again they have modelled the problem as a PERT network, but this time they use a genetic algorithm to find better solutions in the network instead of a local search algorithm. This approach resulted into feasible

solutions for the problem. Moreover, the solutions found by the algorithm were evaluated quite well from a practical point of view.

He et al [19] describes a shunting problem for freight trains. Trains arrive at a shunt yard with freight carriages with different destinations. These carriages have to be combined to form new departing trains, with only carriages for the same destination. The problem exists of two sub problems: the arriving trains have to be classified (i.e. determine which carriage has to go to what location) and departing trains have to be assembled with carriages that have already been classified. For both the classification and assembling job, timeslots are reserved in which one train can be classified / assembled. Because exact arrival and departure times are not that important with freight transportation, He introduces fuzzy start points for the classification timeslots and fuzzy departing times for the departing trains. This means that although there are preferred starting times for classification and departing times for trains, finding a solution where classification/departing starts a bit earlier or later is allowed. In his model He tries to find an optimal processing order for arriving and departing trains, where the number of departing carriages is maximized while maintaining an on-time service as much as possible. To find this optimum order, He implemented a genetic algorithm. The introduction of fuzzy times was quite promising, when compared to an early study from the author where service times were hard constraints [18].

Di Stefano and Koči [30] describe the shunting problem as the problem of minimizing the number of tracks used in a shunting yard while no train is blocking a departing train on the yard. Track capacity or train lengths are not taken into account. In their research four different shunt yards are discussed. The single input, single output (SISO) shunt yard; where trains can approach the shunt yard from one side and leave from the other side. The double input, single output (DISO) yard; trains can approach the yard from two sides, but only leave the yard from one side. The single input, double output (SIDO) yard; trains can leave the yard from two sides and approach from one side. The double input, double output (DIDO) yard; the shunt yard can be approached and left from both sides. For the SISO yard the authors have applied a method based on the graph coloring problem; finding the minimum number of colors to color the vertices in a graph in such a way that no adjacent vertex has the same color. In this case they want to find the minimum number of tracks by constructing a graph based on the arrival and departure sequences of the trains. A vertex represents a train with an arrival and departing time, adjacent vertices mean that these two trains cannot be parked at the same track. The DISO, SIDO and DIDO problems are much more complicated and a valley hypergraph (i.e. a graph in which edges may connect more than two vertices) is introduced to solve the problem. In Cornelsen and Di Stefano [10] the same approach is used but this time also a cyclic timetable (i.e. a timetable in which the situation repeats itself each period) is taken into consideration.

#### **2.4.2 The shunting problem at Netherlands Railways**

In the search for a decision support system for shunt planners, Van Wezel [33] uses a task oriented approach. The goal of this system is not to create or optimize the shunting schedule, but to support the planners that make these schedules. He tries to solve the

problem by looking at the way planners currently solve the scheduling problem by hand. This resulted in a hierarchical task structure consisting of four different tasks; combining incoming with outgoing trains, assigning tracks, determining routes and finally assigning staff. For his research Van Wezel built a prototype planning support system in which each task is solved separately, applying algorithms and methods found in the literature. At each step the planner can see the outcome in a graphical user interface in which he can adjust decisions made by the system. Each adjustment will automatically be analysed by each step and for example return new routes when a train is put at a different track. At the time of publication of Van Wezel's article, the prototype was still being tested. He concludes his research by saying that planning algorithms shouldn't be build to solve entire planning problems, but to solve subtasks for planners.

Freling et al. [14] were first to introduce the train unit shunting problem (TUSP), the problem of matching arriving and departing train units and parking these on a shunt track. They divide the problem into two smaller problems; first they match the arriving units with the departing units, minimizing the shunting effort by keeping units together as much as possible. Secondly they park these matches on a shunt track in such a way that the capacity is never exceeded, units with an earlier departure time are never parked behind a unit that departs later and the cost is minimized (e.g. routing costs). For the matching procedure they have formulated a mathematical model, which is solved by a MIP solver. To assign the units to shunt tracks they have used a column generation heuristic. The matching problem is quite easy to solve with a MIP solver and for the track assignment problem the column generation heuristic is able to find a near optimal solution in an acceptable computation time in most of the cases. The motivation for this 2-step approach is mainly due to the increase of complexity when solving the problem in an integrated matter. But also the ability to interact during the planning process is an advantage; planners can modify the plan right after the first step. The drawback of this approach is that the model will probably not generate a global optimum and there may not always be a feasible solution (e.g. when not all the matches in the first step can be parked).

In the work of Lentink et al. [22] the TUSP problem is decomposed in four parts, and a four-step algorithmic solution approach is developed to assist planners in creating parts of the shunt plan. The four steps are: matching arriving to departing train units, estimating routing cost, parking train units on shunt tracks and finally solving the routing of train units. For step 1 and 3 the same approach as Freling et al. [14] is used, with the difference that the routing cost that is used to minimize the parking cost in step 3 is calculated in step 2 instead of using predefined values. In order to calculate the routing cost and determining the route in step 4, a model of the infrastructure is introduced in which shortest paths can be found and where possible conflicts are detected. This model is represented as a graph where all tracks and switches are represented as nodes. An Occupied Network A\* search algorithm is used to find optimal paths in the network. This four-step approach was able to generate feasible shunting plans which have been validated by planners at NS. Although the generated plans still needed to be modified by the planners, they thought that such an approach could support them in creating shunt plans.

The research of Abbink [1] focuses on dealing with constraints. He tries to solve the TUSP by applying Constraint Satisfaction Programming (CSP) and Domain Reduction. To motivate this approach, Abbink sums the criteria he thinks are important for scheduling techniques; provide (near) optimality, handle large cases, involve a heuristic approach, solve the problem integrally. This last point is interesting, because most other researchers tried to solve the TUSP in separate steps. The CSP technique proved to find feasible solutions within 1 minute. The objective function still only includes this feasibility test so a solution may not be the suitable according to the planners. Including a better objective function that also takes planners' preferences into account such as minimizing the shunting effort would be the next step for this approach. Also the degree of details is not yet satisfactory, which means that the model can generate "feasible" solutions that in practice are still violating constraints that were not taken into account.

In Haijema et al. [17] a heuristic approach inspired by dynamic programming is introduced for the TUSP. In this approach, first the departures are positioned by a blueprint algorithm, then arrivals are matched to the departures by a matching algorithm. The blueprint algorithm determines from which track train units are collected to form a departing train. This algorithm tries to avoid crossings, minimizes divisions (i.e. collecting units from different tracks for the same train), clusters train units of the same type and minimizes the shunting effort. The matching algorithm matches the arriving units to departing units by finding a good positioning of the arrival units on the blueprint generated by the first algorithm. This two step approach cannot guarantee that a solution exists, because the blueprint position can result in arriving units that can not reach their matched departing unit. However, for the experiments in this research such a situation did not occur and results were quite promising with computing times of less than a second. Still more testing of the heuristic is needed on different problems taken from practice before a good evaluation can be given.

Kroon et al. [21] introduce an integrated approach for TUSP by combining the matching and parking sub problems into one model. This integrated approach makes the problem much more complex. The main bottleneck of the model is the large number of crossing constraints, one for each potential crossing at each shunt track. When applied to the real-life case of Zwolle, the model contained over 400.000 constraints and the MIP solver of CPLEX was not able to find a feasible solution within a reasonable amount of time. To tackle the problem, they have strengthened the crossing constraints by aggregating them into clique constraints and tried to reduce the number of crossing constraints. These improvements reduced the number of constraints greatly, and feasible solutions were found in the case of Zwolle. However, when dealing with larger problems, computation times can increase rapidly and moreover it is quite complicated to extend the model with special cases that arise in real-life.



## Chapter 3

# The MIP Solution

In this section we describe the approach that is currently in development at NS. Based on the research in the Rintel project, NS has chosen to go for a Mixed Integer Programming (MIP) approach in which both the matching as well as the parking problem is solved integrally. This approach is a result of the work of Lentink, Kroon and Schrijver [22, 21]. First we will describe the mathematical model, followed by an evaluation. In the remainder of this thesis, we will refer to this model as the MIP model which will be used to compare our results.

Mixed integer programming is a technique for optimization of an objective function, subject to a set of constraints. By changing the decision variables without violating any of the constraints an optimal solution is tried to be found. Finding an optimal solution in a MIP model can be done with several techniques, however when the problem becomes too large there is no guarantee that an optimal solution can be found. Solvers try to approach the optimal solution in these cases. The most commonly used technique is the CPLEX solver [20] developed to solve linear programming models and later extended with a MIP solver. This solver first finds a lower bound with linear relaxation and using a branch and bound technique a good (close to the lower bound) integer solution is tried to be found. The model developed at NS also uses a CPLEX solver.

In the next section the “basic model” is explained in detail to give an idea of the complexity of the MIP model. This basic model only includes LIFO shunt tracks and two objectives. In the current approach this model is extended with free tracks and more location specific objectives. These improvements on the basic model are discussed and in the following section the “extended” model is evaluated with the shortcomings that arise with this model. In the final section of this chapter, based on the evaluation, we formulate the requirements for our new approach that we introduce in chapter 4.

### 3.1 The Basic MIP model

The following model describes a basic model for the TUSP formulated by Kroon et al [21]. We start with some notation:

$S$	the set of shunt tracks at which train units can be parked.	
$c_s$	the length of shunt track $s$	
$T$	the set of train units that arrive or depart	$T = \{1, \dots, n\}$
$T_+$	the set of train units that arrive	$T_+ \subset T$
$T_-$	the set of train units that depart	$T_- \subset T$
$Y$	the set of different train unit types	
$\tau_t$	train unit $t$ is of type $\tau_t$	
$l_t$	length of train unit $t$	
$i_t$	train unit $t$ arrives/depart in train $i_t$	
$L$	the set of pairs of an arriving train unit and a later departing train unit of the same type	
$X$	the set of potential crossings of pairs in $L$ at a LIFO track	

Next we introduce the decision variables:

$$\begin{aligned}
 z_{t,s} & \begin{cases} 1 & \text{if train unit } t \text{ is parked at or retrieved from shunt track } s \\ 0 & \text{otherwise} \end{cases} \\
 x_{t,u,s} & \begin{cases} 1 & \text{if arriving train unit } t \text{ is matched with departing train} \\ & \text{unit } u \text{ and parked at shunt track } s \\ 0 & \text{otherwise} \end{cases} \\
 b_{t,s} & \begin{cases} \text{the length of the train units at shunt track } s \text{ immediately} \\ \text{after the arrival or departure of train unit } t. \end{cases} \\
 d_t & \begin{cases} 1 & \text{if train units } t \text{ and } t+1 \text{ are related to the same train} \\ & \text{and are parked at or retrieved from different shunt tracks} \\ 0 & \text{otherwise} \end{cases} \\
 m_{\tau,s} & \begin{cases} 1, & \text{if at least one train unit of type } \tau \text{ is parked at shunt} \\ & \text{track } s \\ 0, & \text{otherwise} \end{cases} \\
 n_s & \begin{cases} \text{the number of types } \tau \text{ exceeding 1 parked at shunt track } s. \end{cases}
 \end{aligned}$$



In this basic model, the objective function has two objectives;

1. minimize the number of different train types parked on the same track  $n_s$ .
2. minimize the number of arrival and departure breaks  $d_t$

The first objective adds robustness to the solution; units of the same type are interchangeable which means that we do not have to care about the order of the units if each unit is of the same type. The second objective minimizes the shunting effort; if an arrival or departure is broken, it means that you have to park/retrieve units from the same train to/from different shunt tracks. The basic MIP model reads:

The objective function:

$$\min D \sum_{t \in T} d_t + N \sum_{s \in S} n_s \quad (3.1)$$

Subject to:

$$\sum_{s \in S} z_{t,s} = 1 \quad \forall t \in T \quad (3.2)$$

$$\sum_{u \in T_+ : (t,u) \in L} x_{t,u,s} = z_{t,s} \quad \forall t \in T_+, s \in S \quad (3.3)$$

$$\sum_{t \in T_+ : (t,u) \in L} x_{t,u,s} = z_{u,s} \quad \forall u \in T_+, s \in S \quad (3.4)$$

$$x_{t,u,s} + x_{v,w,s} \leq 1 \quad \forall s \in S, \{(t,u), (v,w)\} \in X \quad (3.5)$$

$$b_{t-1,s} + l_t z_{t,s} = b_{t,s} \quad \forall t \in T_+, t > 1, s \in S \quad (3.6)$$

$$b_{t-1,s} - l_t z_{t,s} = b_{t,s} \quad \forall t \in T_-, t > 1, s \in S \quad (3.7)$$

$$0 \leq b_{t,s} \leq c_s \quad \forall t \in T_+, s \in S \quad (3.8)$$

$$z_{t-1,s} - z_{t,s} \leq d_t \quad \forall s \in S, t \in T, t > 1, i_t = i_{t-1} \quad (3.9)$$

$$z_{t,s} \leq m_{\tau_t,s} \quad \forall s \in S, t \in T \quad (3.10)$$

$$\sum_{\tau \in Y} m_{\tau,s} \leq n_s + 1 \quad \forall s \in S \quad (3.11)$$

$$z_{t,s} \in \{0, 1\} \quad \forall t \in T, s \in S \quad (3.12)$$

$$x_{t,u,s} \in \{0, 1\} \quad \forall (t,u) \in L, s \in S \quad (3.13)$$

The objective function tries to minimize the two objectives, the weights  $D$  and  $N$  give the importance of one objective over the other. Constraint (3.2) state that each train unit must be parked at exactly one shunt track. Constraint (3.3) states that each arriving train is matched with a departing train unit and is parked at a shunt track. Constraint (3.4) states the same for departing train units. Constraint (3.5) prohibits crossings of train units, saying that 2 matched pairs that potentially cross each other (exist in set  $X$ ) can not be parked at the same shunt track. Constraints (3.6) and (3.7) register the used capacity of the shunt track immediately after an arrival or departure. Constraint (3.8) ensures that the total capacity of the shunt tracks never exceeded. Constraints (3.9), (3.10) and (3.11)

are used for determining the right values of the decision variables in the objective function. Constraints (3.12) and (3.13) are integer constraints for the  $z_{t,s}$  and  $x_{t,u,s}$  variables. As mentioned before, the current model is far more advanced than this basic model. New constraints have been added that represent the shunting problem in more detail and existing constraints are strengthened. The model takes more objectives into account, namely;

- |            |   |
|------------|---|
| Minimize : | - arrival and departure breaks            |
|            | - different train types on the same track |
|            | - shunt moves                             |
| Maximize : | - park on preferred tracks                |
|            | - park on empty tracks                    |

### 3.2 Evaluation of the MIP model

In this section we will evaluate the MIP model. As you may have noticed, the MIP model does not include all the decisions a shunt planner has to make (see chapter 2). The routing of train units from and to the platform tracks, shunting between shunt tracks, the time a shunt activity takes place and the scheduling of the crew are not included in the model. The current opinion is that these decisions can be seen as different problems which can be solved once you have a good shunt schedule. However some of these decisions do influence the shunt schedule.

In collaboration with the people developing the model and local planners that will finally have to use such a planning tool, the performance of the model has been reviewed. The model is tested on shunt problems at different shunt locations in the Netherlands and the outcome is discussed with the local planners at those locations. The results proved to be quite promising at “less complicated” locations. At these locations the infrastructure is not so complicated and/or the number of train units that need to be shunted is not too large. For example in the case of shunt location Hoofddorp, the model can generate a complete shunting schedule for one day which the planners reviewed as a good shunt schedule. The computation time is acceptable; round 10 to 15 minutes. Also at the shunt location Alkmaar, the model was able to create a complete shunt schedule that could be used in practise. Combining more days in one run could be useful in the case of the weekend schedule, but results at Hoofddorp were far less acceptable and the computation time increased dramatically when multiple days were optimized together. An allowed solution was found after 2,5 hours and after 14 hours the result was still not good enough to be accepted by the planners. This problem can be overcome by splitting the weekend in smaller parts, and using the output of one day as input for the next.

At the more complicated locations, other problems arise that are imposed by the simplification of the shunting problem. Two major shortcomings came forward for which a solution is not yet at hand. One of these shortcomings is the assumption that the time of a shunt activity is directly after or just before the train’s arrival/departure at the platform. In practice, planners often vary the shunt time to overcome infra capacity problems and sometimes even save shunt movements or realize a better unit order at the shunt yard. In

the case of the shunt location Rotterdam we saw several shunt schedules where this variation in time did affect the quality of the shunt plan. Including time into the MIP model would mean that for each arriving or departing unit, a range of shunt time possibilities should be included. This would make the problem tremendously larger and making the MIP solver unable to find any solution. To illustrate; lets take a small case with 10 arriving train units and 10 departing train units. In total there are 20 shunt activities, 10 for the arriving units from their arriving platform to a shunt track, and 10 for the departing units from a shunt track to their departing platform. If we would include a choice of two different moments in time for each shunt activity, it would mean that the search space becomes  $2^{20} = 1048576$  times larger. Now we have only included 2 moments in time, which still does not come close to the range of choices a planner has in real life.

The second shortcoming is the assumption that each arriving unit that departs at a later time can only be shunted from its arriving platform to a shunt track and back to its departure platform. In practise however, units are now and then shunted from one shunt track to the other before they are shunted to their departure platform. Adding this kind of freedom is virtually impossible; in theory each arriving unit could be shunted innumerable times before it is shunted to its departing platform. You could simplify this by allowing only 1 extra shunt activity, but even this will make the problem enormously more complex, because for this extra shunt activity you also have to include a range of possible moments in time at which this extra activity could take place.

### 3.3 Requirements for a new approach

As we have mentioned in the introduction, the goal of this research is to introduce a new approach for the TUSP. This new approach should include some new level of detail to the shunting problem, details the MIP approach is currently not able to include. In the evaluation of the current approach, two shortcomings came forward regarding this level of detail. In our new approach we want to include one of those, namely; flexible shunt times. In our opinion including extra shunt possibilities for each unit makes the problem unnecessarily large. In practise these moves are normally only performed when train units need to be washed or when the schedule needs to be changed because of delays or if tracks are out of service. If such events occur, we think that the local planner should adjust the shunting plan by hand instead of creating a model that includes all these possibilities. By including flexible shunt times we think that we can realise better shunt plans that fit more with the plans local planners build currently by hand. If we want to include flexible times, it will mean that our approach must be able to find good solutions in a much larger search space. However we do not want an approach that takes forever to compute a single shunt schedule, thus it should also be able to find a solution in this much larger search space in a reasonable amount of time. Besides this extension we want our approach to be able to generate plans as good as, or even better than, the MIP model and be able to generate good plans when the problem becomes larger (e.g. over a longer time period). By good plans we mean plans that need little or no adjustments. If we would generate plans that need a lot of adjustments, planners will probably ignore the decision tool and build the schedules in the traditional way. The MIP model has already proven to provide good plans

in a few cases, so therefore we will use the MIP approach as a benchmark when possible.

The remaining question is: how much time is considered to be reasonable? Shunt plans are normally (e.g. if no unexpected disturbances occur) finalised a few days before the actual planned day, but the initial schedules are build a few months in advance. As we have mentioned in section 1.1, the aim of NS is to postpone the scheduling process to a few days in advance. In this new situation planners have 3 to 4 days to complete the entire schedule. The reasonable amount of computation time for a scheduling tool depends on the quality of the provided schedules. In theory, if the tool would always provide quality schedules for which no adjustments need to be done, the computation time could take up to 4 days, after which the computed schedule directly comes into operations. In practice you probably do not want to give the control entirely out of hands, and human interference will always be necessary even if it was only to examine the computed solutions. However the aim of the decision tool in development is not to provide finished shunt schedules, but to support the planners with a good initial solution on which adjustments still can or have to be made. Providing completely finished schedules is currently considered to be impossible because of the complexity of the entire shunt process (including routing, crew scheduling, cleaning, ect..). The provided solution by the decision tool will probably need adjustments from the local planners, and after they fit in all the other elements, they might want to use the tool to recalculate the plan with new input. Therefore in our case, the maximum amount of computation should be within a few hours, let's say 3 hours, so that planners have enough time to evaluate, adjust and maybe recalculate the plans provided by the tool.

The requirements for the new approach are:

1. Introduce flexible shunt times for each shunt activity.
2. Compute solutions in a reasonable amount of time (max 3 hours)
3. Compute high quality shunt plans (little adjustments needed)

## Chapter 4

# A Metaheuristic Solution

In the previous chapter we described the model that is currently under development at NS. Although the results of this study are quite promising, and further research along this path could be interesting, we are going to introduce a new approach for the TUSP that tries to tackle one of the problems the current approach is yet unable to solve, namely introducing flexible shunt times.

We formulated three requirements that this new approach should be able to apply. The first requirement states that we will have to introduce flexible shunt times for each shunt activity. That means our approach will need to solve much larger search spaces than the MIP approach is currently solving. The second requirement states that computation times should be reasonable (max 3 hours). So our approach does not only have to solve larger search spaces, this should also be done in a limited amount of time. The third requirement states that the model should provide good shunting plans. Since we will use the MIP approach as benchmark, let us discuss the strength of this approach. In literature several approaches were found where the optimization was divided into two steps. First find suitable matches, then park these matches on the shunt track. Eventually an approach where these two elements were solved integrally was chosen over the others, because dividing the problem gave too much complications (e.g. infeasible solution, bad plans). Therefore our approach should also solve the problem integrally.

In the past 20 years a lot of successful applications in Operations Research using metaheuristics have been reported in the literature. Techniques such as simulated annealing, tabu search and genetic algorithms have proven to find high-quality solutions to many real-world optimization problems [28]. In literature a lot of researchers have successfully implemented a genetic algorithm (GA) for scheduling problems such as the job-shop scheduling problem [11, 16, 23] and the project scheduling problem [25, 6]. GA's are able to find good solutions even when the problem instances become very large. In general genetic algorithms are known for their ability to find very good approximate solutions in very large search spaces in many fields of operation research [35]. Supported by other research in the field of scheduling, and because we will need a technique that can handle very large search spaces to handle the time extension, we have chosen to implement a genetic algorithm to

solve the Train Unit Shunting Problem. In the remainder of this chapter we will describe this GA approach, starting with the formulation of the TUSP.

## 4.1 TUSP formulation

This TUSP formulation follows from the description given in chapter 2. In our model we are going to optimize four different decision variables, two for each arriving unit that needs to be shunted to the shunt yard:

1. Select a shunt track for each arriving unit.
2. Select the shunt time for each unit to its shunt track.

And two for each departing unit that has to be shunted from the shunt yard

1. Select a parked unit for each departing unit.
2. Select the shunt time for each unit to its departing platform track.

With arriving units that “need to be shunted”, we mean the arriving units that stop their service after they arrive at the station, and thus have to be parked for a period of time. With departing units that “need to be shunted”, we mean the departing units that start their service at the station and thus a parked unit is needed to fulfill this service. For other train units that will make a small stop at the station until they continue their service or train units passing by the station without a stop, no decision has to be made, however they have to be taken into account because they do utilize the track capacity at the platform tracks for a period of time. The GA will try to optimize these variables taken the restrictions and objective presented in Chapter 2.2 into account.

As you may have noticed, we did not include all the decisions a shunt planner has to make into our model (see Chapter 2.2). The routing of train units from and to the platform tracks and shunting between shunt tracks is considered to be out of the scope of this research. In our opinion the routing problem and crew scheduling can be seen as different problems which can be solved once you have a good shunt schedule. As we explained in the previous chapter, including extra shunt moves would make the problem unnecessarily large. In the next section we describe the GA that implements this formulation.

## 4.2 Genetic Algorithm

The genetic algorithm is a search technique inspired by Darwin’s theory of evolution by natural selection [12]. In biology, characteristics of a population are passed on from one generation to the next. The characteristics of an individual are stored in his genes, which he will pass on to his offspring. The process of evolution is the change in the inherited characteristics. Darwin’s theory says that natural selection drives the evolution process to populations with characteristics best-suitable for their environment. In a population, individuals with advantageous characteristics are more likely to survive and thus to reproduce,

resulting in an offspring generation with more individuals that inherited these characteristics. In each generation small changes in genes (i.e. mutations) can occur. Over many generations, because of this natural selection process, positive changes will become more common in a population while negative changes become rare.

The genetic algorithm uses these principles to find (near) optimal solutions to optimization and search problems. A candidate solution (individual) in a GA is represented as an array of bits or values, called a chromosome. At the start of the algorithm, a population of individuals is created with randomly generated chromosomes. Each individual has a fitness value which represents the performance of the solution. To calculate this fitness the chromosome of the individual is used as input to a model or function and a value is returned that tells how well the candidate solution performed. A new individual is produced by selecting a pair of “parent” individuals from the population based on their fitness. The new “offspring” individual is created by combining the chromosomes of both parents into a new chromosome. For this process different types of crossover operators can be used. The most basic one is the one point crossover; a random point in the parents chromosomes is chosen, the chromosome of the offspring is formed by the left part of one parent’s chromosome and the right part of the other parent’s chromosome. To keep diversity in the population and prevent local search, some offspring individuals will “mutate”, meaning that a part of the chromosome is randomly changed. The fitness of the individuals of this new offspring population is then calculated and a new generation is created consisting of the best parent and offspring individuals. This process continues until a fixed number of generations is reached, a solution is reached that satisfies some criteria, or a number of successive generations no longer produced individuals with a better fitness.

The main issue of using a genetic algorithm is to formulate a possible solution to the problem into a chromosome and create a fitness function that can evaluate this solution. The chromosome in our approach will represent a possible shunt schedule consisting of the two decision variables for each arriving train unit and the two decision variables for each departing train unit. For the fitness evaluation of these possible shunt schedules we are going to simulate the actual process of shunting instead of trying to formulate the entire process in a mathematical way. With simulation we can represent the problem in much greater detail and allow typical shunting events that do occur in the real world. Simulation can also help to get a better understanding of the problem because you can easily visualise the actual shunting process. Moreover with simulation it is quite easy to introduce delays and disturbances in the shunting process. In practice planners have to deal with these problems every day, so in future research it could be interesting to measure the performance of shunting schedules in the event of these delays or other disturbances.

Another difficulty is to select the appropriate parameters for the evolution process in such a way that the algorithm is not searching towards a local optimum. The parameters that can be changed by the user in our implementation are:

**Number of generations** After how many generations does the search stop

**Population size** The number of individuals in the population

**Crossover rate** What is the chance that crossover occurs

**Mutation rate** What is the chance that mutation occurs

**Elitism** The number of individuals of the parent population that is taken to the next generation

These parameters should be chosen carefully and are, unfortunately, problem dependent. This means that there is not one good set of parameters that can always be used.

To outline our approach: the genetic algorithm will try to find good shunt plans by adjusting the decision variables while a simulation model will evaluate each solution created by the GA. Figure 4.1 gives a schematic diagram of the steps of our approach.

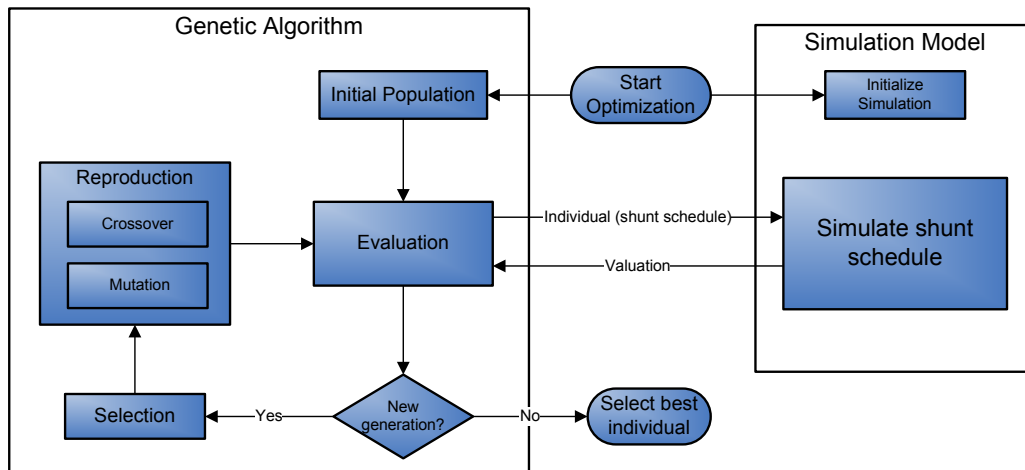


Figure 4.1: The Genetic Algorithm diagram

In the remainder of this section we explain how we have implemented the shunting process into this chromosome representation and discuss the different steps in the algorithm in more detail. In the a later section the simulation model used for the evaluation is discussed.

#### 4.2.1 Chromosome representation

An individual represents a candidate solution to the problem you are trying to optimise. In our case we want to optimise a shunting schedule so we need to fit this schedule into the chromosome of an individual. As mentioned before, a chromosome must be represented as an array of bits or values. The shunting schedule consists of four different types of decision variables explained in the previous section. In biology genes are small parts in the chromosome of the organism that holds specific information of one characteristic of that organism. For example the color of the eyes is stored in one gene in the chromosome. The information in a gene is coded as a sequence of different alleles. In our chromosome representation we divided the chromosome into 4 different genes, each gene representing one of the different decision variables. Two genes for the arriving units and two genes for the departing units. Each allele in the gene represent either a arriving unit or a departing unit.



**Arriving units** The first gene represents the shunt track assignment for arriving trains. Each place in the array (allele), represents an arriving train unit that needs to be shunted. The value of the allele represents the track this unit is shunted too. The order of the alleles in the array is equal to the order in which the units arrive at the station. In the case of a *Free track*, a track that can be approached from both sides, two numbers are used that represent the same track. One number if the arriving unit enters the track from the A-side and one number if the arriving unit enters the track from the B-side.

Arriving unit number	1	2	3	4	5	6	7	8	9	10	
Track number	2	5	5	0	1	1	2	3	4	2	}Parking Gene

In the example above the first arriving train unit that needs to be shunted will be parked at shunt track 2, the second at shunt track 5, the third will also be parked at track 5 and so on. Note that trains can arrive in a composition of multiple units. If units in the same composition have the same track assigned, the composition is shunted in one movement to the shunt track and is counted as 1 shunt move. The range of these values is between 0 and the total number of shunt tracks.

The second gene represents the shunt times for the arriving units. Once again each allele represents an arriving train unit that needs to be shunted. The values of the alleles describe the number of minutes arriving units will have to wait at the station before they are shunted to a shunt track. In practice units should always wait a couple of minutes before they are shunted. This minimum waiting time can be set by the user and is added to the value of the allele to determine how long a unit will have to wait after its arrival at the station.

Arriving unit number	1	2	3	4	5	6	7	8	9	10	
Minutes after arrival	4	0	10	5	0	8	0	0	5	5	}Wait Time Gene

Let's say that the minimum wait time is 2 minutes. In the above example the first arriving train unit that needs to be shunted will be shunted to its shunt track 4+2 minutes after it arrives at the platform. The second value is 0, which means that the second arriving unit will be shunted 2 minutes after its arrival at the platform.

**Departing units** The third array represents the assignment of parked units to departing trains. In this case, each allele represents a departing train unit that needs to be shunted to its departing platform. The order of the alleles in the array is equal to the order in which the units depart at the station. The value of the allele represents the arrival number of the unit this departure is matched with.

Departing unit number	1	2	3	4	5	6	7	8	9	10	
Arrival unit number	2	1	3	6	4	5	7	8	10	9	}Matching Gene

In the above example, the first departing unit is matched with the second arriving unit. Note that in this case, the values of the array are somewhat restricted. The values need to be unique in the array because you can't assign one arriving unit to two departing units and you can only assign arrived units to departing units if the arrival is before the departure. Moreover, each arriving unit should be in the sequence, meaning that there are exactly as many arrivals as departures at the location. This is normally the case in shunting schedules.

The fourth gene represents the shunt times for departing units. Each allele represents a departing train unit, the same as the third gene. The values of the alleles represent the number of minutes before the departure time at which departing units are shunted to their platform. The minimum wait time of the second gene also applies for this gene.

Departing unit number	1	2	3	4	5	6	7	8	9	10	
Minutes before departure	5	0	0	10	10	4	3	0	2	2	}Early Time Gene

In this example the first unit that will depart is shunted to the departure platform 5+2 minutes before the departure time of the train. Zero means that we only use the minimum wait time, which in this case would mean that the unit is shunted 2 minutes before its departure.

**Shunt time possibilities** Theoretically the range of the values of the alleles in the second and fourth gene could be infinite, however we downsized the range to a user specified number of possible moments for each unit. For each unit these moments are chosen based on the events in the timetable. In our model, the only interesting moments to shunt a unit besides just after/before its own arrival/departure, are just before, at the same time, or just after the arrival/departure of an other unit. These are the moments the order and capacity changes at the tracks or shunt moves can be combined (i.e. shunting multiple units from different trains to/from the shunt yard at the same time). For example we have a shunt location with the timetable shown in table 4.1. Let’s give each unit 10 possible shunt moments. In table 4.2 we show the shunt possibilities for the first arriving unit, and the last departing unit. For arriving units we add the shunt times for each upcoming event to the possibilities and 2 minutes after each event. For departing units we do the same, but for the events before this departing event.

Table 4.1: 8 timetable events

Arrive/Depart	A	A	D	A	D	A	D	D
Time	10:00	10:20	10:27	10:50	11:02	11:10	11:20	11:32

Table 4.2: 10 shunt time possibilities

Unit 1	Minutes after arrival	0	20	22	27	29	50	52	62	64	70
Unit 8	Minutes before departure	0	12	14	22	24	30	32	42	44	65

**The complete shunt schedule** We have just described the four genes that together form a complete shunt schedule, table 4.3 illustrates such a schedule. Each color represents one and the same physical unit and its shunt schedule. A train unit that is shunted, always follows the same 4 steps: the unit arrives at the station, is parked at a shunt track, is retrieved from a shunt track and finally departs at the station. In this example the first train unit that arrives at the station will be shunted to shunt track 2, four minutes after its arrival. This same unit is selected for the first departing unit and will be shunted 5 minutes

before its departure time to the platform. The second unit that arrives at the station will be shunted to shunt track 5 directly after its arrival. This unit is selected for the fifth departing unit at the station and will be shunted to its departing platform 12 before its departure time.

Table 4.3: A complete shunt schedule represented in a GA chromosome

Arriving unit number	1	2	3	4	5	6	7	8	9	10	
Track number	2	5	5	0	1	1	2	3	4	2	Parking Gene
Minutes after arrival	4	0	5	10	7	7	8	10	0	5	Wait Time Gene
Departing unit number	1	2	3	4	5	6	7	8	9	10	
Arriving unit number	1	4	3	5	2	6	9	8	10	7	Matching Gene
Minutes before departure	5	0	0	10	12	0	5	8	10	0	Early Time Gene

### 4.2.2 Initial population

The first step in the algorithm is creating an initial population. This initial population consists of  $x$  individuals, where  $x$  is the population size chosen by the user. For each individual we have to choose the values for the alleles in the four genes described in the previous section. For each allele in the parking gene a random number within the range is chosen. For the matching gene the values of the alleles are randomly chosen constrained by the restriction that each value can only appear once and the restriction that the departing units can only be matched with arriving units that arrive before the departure of the departing unit. For the second and fourth genes all the values of the alleles are initially set to 0. In most cases, shunting just after/before the arrival/departure will be the best option. By modifying the shunt times in the mutation phase of the evolution process we hope to find the special cases where adjusting the shunt times can be beneficial.

### 4.2.3 Fitness evaluation

To compare the individuals with each other, we need a method that can evaluate the solutions provided by the individual's chromosome. As we mentioned in the introduction of this chapter, we are using a simulation model to evaluate the performance of a solution. Given the solution provided by the chromosomes the simulation model simulates the entire process and registers all the movements and events that occur. We want to evaluate the solutions based on the preferences we specified in section 2. A solution that applies these preferences better is considered to be a better solution and thus has a better fitness. However we also need to take the restrictions into account. The GA has no information of the problem it is optimizing so it does not know whether a solution is feasible (i.e. satisfying all the restrictions) or not. The only information that is provided during its search is the fitness of each solution. In order to handle these restrictions, we have to introduce penalties. The fitness of a solution is downgraded with a certain penalty for each restriction it is violating.

## Constraint handling

In the literature, a lot of constraint handling methods have been suggested. Deb [13] for example proposes an approach where he separates the objective functions from the penalties. He considers a solution without any penalties to be always better than a solution with penalties. Although this is true for the final solution we are trying to obtain (i.e. an infeasible solution can not be used in the real world), this method is more likely to find a local optimum instead of the global optimum. For example there may exist an infeasible solution in the population that is close to the global optimum solution, but if all the other solutions in the population are feasible then this solution is valued as the worst solution and will not be selected for reproduction and eventually it will disappear from the population.

Other approaches combine the penalties and objectives into one fitness function. This means that when a restriction is violated, the fitness of the solution becomes worse. In this case, an infeasible solution that only violates one restriction but performs quite well on the objectives can be graded better than a feasible solution. The problem with such an approach is to determine the appropriate weights for these penalty functions. If the penalties are weighted too low, the optimum value of the fitness functions may not be a feasible solution. On the other hand, if a large weight is used, we get the same situation as in the first approach. Michalewicz [24] tested several sophisticated ways to include penalties into the fitness functions. He concludes that although some of the methods do perform better on typical problems, in general including user-supplied penalty weights is the best overall solution if the weights are well chosen.

In our case we use the user-supplied weights approach. A solution that for example violates a capacity constraint on a shunt track may be very close to an optimal solution by just selecting a different track for one of the units on this track. Therefore we want to keep infeasible solutions that look promising in the population.

## Fitness function

The fitness function will return the fitness of a solution to the GA once the simulation is completed and all the data are recorded. The components of this function are:

- Shunt move (sm)
- Arrival break (ab)
- Departure break (db)
- Not on preferred track (npt)
- Not on empty track (net)
- Wait time (wt)
- Capacity restriction (car)
- Crossing restriction (crr)
- Infra activities restriction (iar)

- Wrong departure order (wdo)

For the definition of these components see Section 4.4. It should be clear that we want to minimize all of the above components. We consider a solution with a lower fitness value to be a better solution.

$$\begin{aligned}
 \textit{Fitness} = & \textit{sm} \times w_{\textit{sm}} + \textit{ab} \times w_{\textit{ab}} + \textit{db} \times w_{\textit{db}} + \textit{npt} \times w_{\textit{npt}} \\
 & + \textit{net} \times w_{\textit{net}} + \textit{wt} \times w_{\textit{wt}} + \textit{car} \times w_{\textit{car}} + \textit{crr} \times w_{\textit{crr}} \\
 & + \textit{iar} \times w_{\textit{iar}} + \textit{wdo} \times w_{\textit{wdo}}
 \end{aligned}$$

The weight( $w$ ) of each component will have to be carefully chosen by the user for each problem. In general you can keep the same set of weights for different instances of the same problem, for example different days in the timetable for the same location. But for different locations a different set of weights should be chosen that applies to the preferences of the planners of that location. Objectives that are more important have higher weights than less important objectives. In general restrictions should always have a higher weight than objectives; hence a solution that is not feasible can not be used. However as we have mentioned before, these weights should not be too heavy in comparison with the weights of the objectives to prevent local search.

#### 4.2.4 Selection

In the selection phase two individuals are chosen to reproduce a new individual. These two “parent” individuals are selected from the population according to their fitness. Individuals with a better fitness are more likely to be selected. In the literature different selection techniques have been proposed. Roulette wheel and Tournament selection are the two most commonly used. The roulette wheel technique selects individuals with a certain chance given by their fitness. With tournament selection 2 or more individuals are randomly chosen and the fittest individual of this group is selected. We have chosen to implement the tournament selection with groups of 2 individuals because we think that this technique gives less fit solutions a better chance to reproduce and thus keeps the population more diverse. Because of the large number of objectives and restrictions, the fitness of different individuals can be far apart. Using the roulette wheel technique will give fit individuals a too large advantage over less fit solutions, because the selection chance is normally distributed according to the fitness of each individual. Still tournament selection gives the worst solutions almost no chance to reproduce, and the worst solution even zero chance, hence it will always lose the tournament. Therefore we give the worst solution of the two selected individuals still a small chance (5%) of winning the tournament.

#### 4.2.5 Reproduction

In the reproduction step the two selected parents from the previous step are used to reproduce a new offspring individual by a crossover procedure. Many different crossover operators exist but the one-point crossover is the most easy one to illustrate; a crossover point is randomly chosen, the offspring’s chromosome is formed by the left part of the first

parent's chromosome and the right part of the second parent's chromosome. In figure 4.2 the one-point crossover is illustrated with a crossover point after the fourth value was used. Other widely used crossover operators include two-point crossover and uniform crossover.

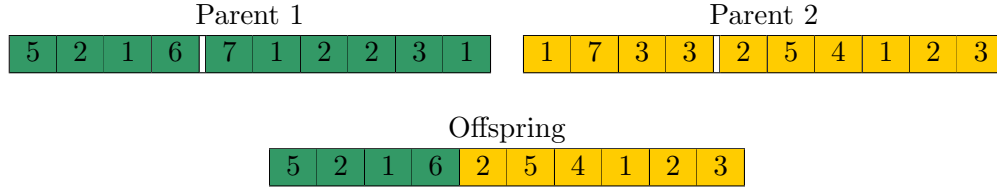


Figure 4.2: One-point crossover

Because in our problem we use 4 different genes in the chromosome, each representing a different type of decision variables, just using a single crossover operator would not be suitable. The matching gene for example is much more constrained than the parking gene, so each should be treated differently. We have decided to divide the chromosome in two parts. One part includes the *parking* - and *wait time gene*, both representing arrival units. The second part includes the *matching* - and *early time gene* both representing departure units (see table 4.4). When the random crossover point is selected, we first check whether it belongs to the first part or the second part. In the case of the first part, we apply the one-point crossover method which will cut both parents chromosomes over both the parking and wait time gene. If the point belongs to the second part, we apply a cyclic crossover method; a crossover method that keeps both the order as well as the uniqueness of each value in mind. The one-point crossover method was selected after a series of tests with different crossover operators (see Appendix A). The second method was chosen for its capability of respecting the restrictions on the matching gene and its successful applications in the literature for permutation problems in the flow-shop problem[7]. With the cyclic crossover, a random

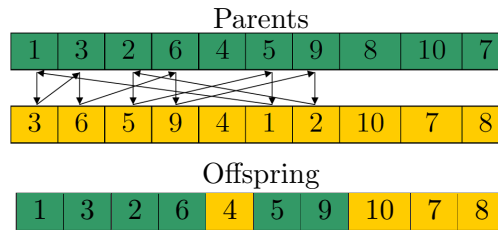


Figure 4.3: Cyclic crossover

allele in the chromosome of the first parent is selected. The value of the allele is used in the offspring. Then we look at the value of the second parent at the same point. We locate the value in the first parent and use this allele in the offspring. Once again we look at the value of the second parent at this point and locate the value in the first parent. This continues until we come back at the starting allele. The alleles in the offspring that have not copied a value from the first parent, will copy the values of the second parent. Figure 4.3 illustrates this procedure.

Table 4.4: The chromosome divided into two parts

Position in chromosome	1	2	3	4	5	6	7	8	9	10	
Parking Gene	1	2	3	1	3	1	3	2	5	4	Matching Gene
Wait Time Gene	0	5	10	0	2	2	0	5	5	0	Early Time Gene
	Arriving units					Departing units					

Let's us now give an example of the entire crossover procedure in our implementation. In figure 4.4 we illustrate what happens if the crossover point is in the first part, at point 3 in the chromosome. As you can see, the crossover point cuts both the parking- and wait time gene at the same point. In the offspring we use the left part of the first parent, and the right part of the second parent. If we select point 8 at the crossover point, thus in the second part, we apply the cyclic crossover on the second part of the chromosome. Copying the matching gene and the early time gene at the same time. The first part of the chromosome is entirely copied from the first parent, see figure 4.5.

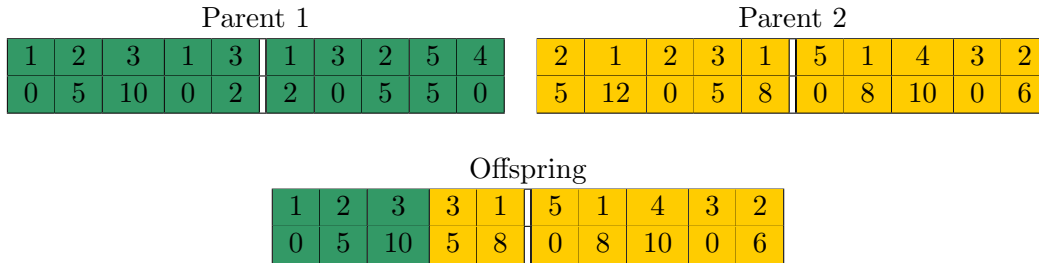


Figure 4.4: Crossover at point 3 using one-point crossover

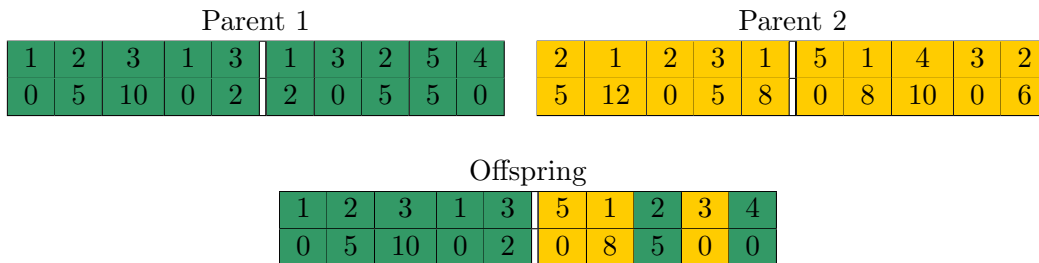


Figure 4.5: Crossover at point 8 using cyclic crossover

We have chosen to combine the parking and wait time gene as well as the matching and early time gene for the crossover procedure because the values represent the same arrival/departure unit. If we would apply the crossover procedure to only one of genes, a lot information can get lost. For example parking an arriving unit at shunt track 2 could only be a good solution if this unit waits 5 minutes before it moves to the shunt track. These specific combinations will get lost during the crossover procedure if the parents have different wait time values and we would only cut the chromosome over the parking gene. This way the offspring only has a part of the parking gene of the first parent, and copies

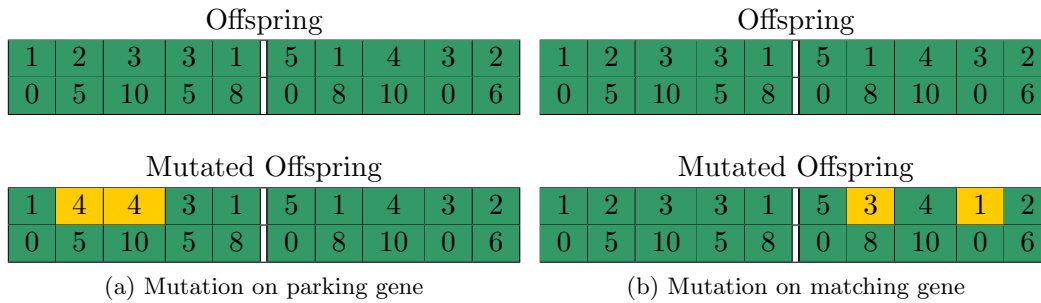


Figure 4.6: Mutation

all the other values of the second parent, including the values of the wait time gene. We have run some test without combining the genes and found out that the GA had a lot more trouble finding specific wait times that are beneficial for the schedule.

## Mutation

To keep diversity in the population, each new offspring has a chance, given by the mutation rate, to mutate. For each of the four genes, a different mutation procedure is formulated. Opposed to the crossover procedure where we combined the crossover procedure of multiple genes, only one gene will mutate each time an individual is selected for mutation. For the parking gene a random allele will change its value to an other random chosen shunt track. But by using this method we can never change the shunt track for an entire composition. Therefore, with a certain chance, two or three adjacent alleles (possibly from the same composition) are changed to the same randomly chosen track. Figure 4.6a shows a mutation on the parking gene, with two adjacent alleles in the gene that are mutated to the same random chosen track. In the matching gene, the values of two random alleles in the gene are swapped. The first allele is randomly chosen, the second is randomly chosen from the alleles that can swap their value with this one: meaning that the units represented by these alleles in the matching gene are of the same subtype and both arrival times of the matched units are before both the departure times of the units represented by the location in the array. Figure 4.6b shows a mutation on the matching gene, where two values are swapped.

For the two genes representing the shunting times, again one, two or three adjacent alleles are selected for mutation. The values are changed to a new random chosen shunt time possibility. Note that we do not chose a random shunt time, but a shunt time possibility. If for example the value is changed to the second shunt time possibility, each allele will change its value to the second shunt time possibility of that unit, which means that the actual shunt times can differ between the alleles. If the adjacent alleles belong to the same composition it will mean the new shunt times will be the same, hence units in the same composition will have the same shunt time possibilities.

In algorithm 1 we show the pseudo code for this mutation procedure.



---

**Algorithm 1** Pseudo code for mutation procedure

---

```
Select one of the four genes
{
  If parking gene
  {
    Select randomly one, two or three adjacent alleles
    Change the values of the selected alleles to a new randomly chosen shunt track
  }
  If matching gene
  {
    Select one random allele
    Select a second allele that can swap its value with the first allele
    Swap the values between the two selected alleles
  }
  If wait time gene or early time gene
  {
    Select randomly one, two or three adjacent alleles
    Change the values of the selected alleles to a new randomly chosen shunt time
    possibility
  }
}
```

---

**Elitism**

For each new offspring the fitness we have produced is calculated. The best  $x$  individuals from the parent population will survive to the next generation and together with the offsprings they will form the new generation. The value  $x$  can be changed by the user, however this value should be at least 1 and at the most 10% of the population size. If too many parents would survive, the population can become very homogeneous, which will slow down the evolution process. If none of the parents survive, good solutions will get lost, and the algorithm will not converge to a near optimal solution.

### 4.3 Alternative Approach

In the previous section we described a method to the TUSP where the entire search space is included into the chromosome of an individual in the genetic algorithm. Although the search space can become quite large, a big advantage is that we are able to find every possible solution within the TUSP formulation. Although genetic algorithms are known for their efficiency in large search spaces, we want to look at a different approach where a part of this search space is solved by a simple heuristic.

The train unit shunting problem is a very restricted problem and one of the limitations of genetic algorithms is handling the conflict between objectives and constraints. A lot of

research has been done on constrained handling for genetic algorithms.. In our approach we try to handle these constraints entirely with penalties that downgrade the fitness of a solution for each constraint that is violated. In this new approach we are going to help the genetic algorithm finding feasible solutions, with the use of a simple rule based heuristic. The most common constraint in the TUSP than can be violated is the crossing constraint. In chapter 3 we described a mathematical formulation of the TUSP and there constraint (3.5) prohibits crossings of train units by saying that 2 matched pairs that potentially cross each other can not be parked on the same shunt track. With “potentially cross each other” we mean that the first arriving unit will depart after the arrival and before the departure of the second unit. In this new approach we will try to reduce the number of possible crossing constraints by excluding the matching variables from the search space. The GA will still provide the parking solution and the shunt time solution for the scheduling problem. Given these solutions a heuristic will select the best suitable matches in such a way that the number of crossings (if any) is minimized. This heuristic can also take the second objective into account: retrieve units from the same composition from the same shunt track.

To motivate this approach we found some successful applications in the literature where the decision variables of the problem were also decomposed and a genetic algorithm was combined with some other search techniques to solve the entire problem. In the work of Palmer and Kershenbaum [27] they try to find optimal trees (i.e. an undirected graph which contains no closed cycles) in a given graph or network. Representing a tree into chromosomes is quite a hard task and creating a chromosome that can represent each possible solution gives an extremely low probability of obtaining any tree. Therefore they developed a method in which the GA gives a bias value for each node, and an algorithm is applied to find the minimal spanning tree over the nodes using the biased cost matrix obtained by the GA. This turned out to be a very effective way, and better solutions were found compared with a good heuristic for this problem at that time. More recent Aickelin has tackled several optimization problems with the use of a so called “indirect” genetic algorithm. In his PhD thesis [2] he introduces the indirect genetic algorithm as a good method for highly constrained optimization problems. In the indirect genetic algorithm, rather than representing the problem “directly”, the chromosome of an individual represents a part of the solution which is given to a decoding heuristic, that generates the complete solution to the problem and its fitness value. This decoding heuristic has some knowledge of the problem and it will try to minimize the number of violating constraints. However, unless a decoder can guarantee feasibility, the use of a penalty functions is still needed since infeasible individuals will still be encountered. In later research Aickelin and Dowsland successfully implemented this indirect genetic algorithm to a nurse-scheduling problem, the problem of assigning nurses to the correct number of day or night shifts [5]. In this indirect approach the GA tries to find the best possible ordering of the nurses which is given to a greedy “decoder” that builds the actual solution. The results were a lot better than in an earlier research of Aickelin and Dowsland [4] in which they try to solve the nurse-scheduling completely with a “direct” genetic algorithm. Using the indirect approach for the set covering problem, Aickelin [3] takes his approach a step further by splitting the search into three phases. First the GA finds good permutations of the rows to be covered along with suitable parameters for the second stage. In this second stage a decoder builds

the solution from the permutation using the parameters provided by the GA. This decoder, opposed to the earlier example, does not try to find the best solutions outright, instead good but further exploitable solutions are built. These solutions are taken to the third phase where a hill-climbing method fully optimizes the solutions. Computational results of this last approach were not yet satisfactory, since more specific heuristics were able to find better solutions. However the results were comparable with other evolutionary algorithms and found with significantly less computation time.

In the research of Cai et al. [9] a genetic algorithm is combined with a linear programming (LP) method to solve nonlinear water management models. Such models try to optimize the production of energy of multiple water reservoirs while satisfying constraints arising from flow augmentation and flood control. They identified a set of complicating variables in the model which, when fixed, render the problem linear in the remaining variables. The GA tries to optimize these complicated variables and a linear programming method solves the remaining problem for each solution provided by the GA. In smaller instances of the problem, a standard non-linear programming solver is more robust in finding good solutions, however when increasing the problem size, the combined GA & LP approach is superior.

#### 4.3.1 A different point of view

In this alternative approach we will try to look at the problem from a planners' point of view. A lot of the decisions that have to be made during the creation of a shunt plan are made by simple human logic which can be hard to translate in mathematical models. The two basic decisions a planner has to make when making a shunt schedule are;

1. Where to park arriving train units.
2. Select parked train units for departing train services

Without any knowledge of the entire problem you could say that only in the first decision a planner is "entirely free" to choose from a range of possibilities; when an arriving train has to be parked, the planner can park the train at all the shunt tracks with enough capacity left. When a departing train needs one or more train units from the shunt yard, then he can only choose from the train units of the same subtype that are parked at the shunt yard. If there are more units to choose from (i.e. more units of the same subtype than needed), he will probably choose the one that will cost the least shunting effort. Basically this means that only the first decision can influence the feasibility and performance of the schedule, hence the selecting decisions will be based on the place where the unit is parked. A good planner has a lot of experience at his location and based on this experience he knows where to park arriving units to create a good shunt schedule. In our case, we try to automate the scheduling process by building a model that can solve the shunting problem in general (i.e. for all possible locations). This model does not have any "experience" at a specific location, and in the case of a new timetable the planners also can not use their experience they have with the previous timetable. In this approach we use the GA to gain this experience, however this time the search space of the GA will only include parking arriving trains on a track and the shunt times, a "simple" rule is used to assign parked units to departing trains.

### 4.3.2 Rule based decision

As we explained in section 4.2.1, the *matching gene* of an individual tells us which arriving unit is matched with a departing unit. In this approach, the matching gene is excluded from the GA. The GA only provides a parking solution for arriving train units and their shunt time to the shunt yard. Also the shunt time for departing units to the platform is provided by the GA, but each time such a shunting event occurs a rule is used to decide which unit at the shunt yard will be used for this departure. This rule does two things:

1. Select the unit with the least shunting effort
2. Try to retrieve all the units in the composition from the same shunt track

The rule has no knowledge of upcoming events (i.e. arrivals and departures) and thus makes its decisions only based on the current situation at the shunt yard. However, if a decision has a negative impact for succeeding decisions in the future, this impact will be shown in the fitness of the solution, causing the GA to park the units at different tracks. See algorithm 2 for the pseudo code of the implementation of this rule. By including this rule

---

**Algorithm 2** Pseudo code for rule based decision

---

```
For all units in departing composition
{
    Determine subtype of unit.
    For each shunt track that stores a unit of the same subtype.
    {
        For each unit of the same subtype
        {
            If unit is in front of shunt track
            {
                If the previous unit was taken from this shunt track
                unit value = 4.
                Else if the next unit subtype is behind this unit on the shunt track
                unit value = 3.
                Else
                unit value = 2.
            }
            Else
            unit value =  $-1 \times$  number of units blocking the path of this unit.
        }
    }
    Choose unit with highest value.
}
```

---

based heuristic in the search space, we are not able to search over the entire search space of the formulated TUSP anymore. Moreover because of the local search characteristics of

the rule we could end up with very bad solutions. In the next chapter we will test both approaches on several cases and will find out whether these shortcomings really affect the quality of the solutions. In the remainder of this research we will refer to this approach as the “rule based approach” and call the other approach the “integral approach”.

### 4.3.3 Adjustments for the GA operators

In this alternative approach the chromosome of each individual will not include the matching gene. Because the values of this gene are highly restricted, we had to implement two different crossover operators for one chromosome. This time, we will only use the one-point crossover operator. We still combine the parking- and the wait time gene when the crossover point is in the first part of the chromosome (see table 4.5).

Table 4.5: The chromosome representation for the alternative approach

Position in chromosome	1	2	3	4	5	6	7	8	9	10	
Parking Gene	1	2	3	1	3						
Wait Time Gene	0	5	10	0	2	2	0	5	5	0	Early Time Gene
	Arriving units					Departing units					

## 4.4 Simulation Model

In this section we will describe the simulation model developed for the evaluation of the shunt schedules provided by the GA. According to the dictionary simulation means: “the representation of the behavior or characteristics of one system through the use of another system”. In our case we want to represent the local shunting process as detailed as possible with the use of a computer simulation model. Computer simulation models can be classified according to two different characteristics. First, the model can be either continuous or discrete. In a continuous model, the state of the system changes continuously and activities can take place at the same time. In a discrete model, activities take place in steps after each other. A special case of a discrete model is the discrete-event model. In such a model, the operations of a system are represented as events that occur at a certain time. In the simulation, the time hops from event to event and, as opposed to a continuous model, the state of the system only changes when such an event happens. Second, simulation models can be divided into stochastic and deterministic models. In a stochastic model, the system evolves unpredictably. Processes can be random and are often described by probability distributions. In a deterministic model, all processes are predictable. Given a certain input, it will always produce the same output.

To describe the shunting process we use a discrete-event representation because the state of the system only changes when trains arrive, depart and are shunted. To optimize our shunt schedule we use a deterministic approach: given the timetable, all events will occur at a predefined time. However, it could be interesting to introduce some stochastic arrival times to simulate delays. In the remainder of this section we discuss the different components, performance statistics and events that are used in the simulation model. And we finish this section with the verification of the simulation model.

### 4.4.1 Components

The components in the model represent the world we are simulating and hold information about the current state of the system. In the shunting world we have trains arriving and departing at stations, trains that are parked at a shunt yard and trains that are moving from one location to the other using the railroad network. The components we use are;

- The train unit: A rolling stock unit that is of a certain subtype, has a length and can move on its own or in a composition with other units of the same type.
- Railroad network
  - Shunt track: A track on which train units can park.
  - Platform track: A track on which trains arrive and depart.
  - Network track: A track that is used to get from one location to the other and parking is not allowed.
- Timetable: A list that holds all the information on arriving and departing trains at the station.
- Shunting schedule: A list that tells which units need to be shunted, to what track and at what time.
- Preferred list: A list that tells the preferred shunt tracks for certain train units.

### 4.4.2 Performance statistic

To evaluate a given shunting schedule, the model stores the following statistics. First the objectives we want to minimize are summarised. The model counts how often the following situations occur. It is up to the planner to value each of the objectives; in different situations / locations an objective can be more or less important than others.

- Shunt move: moving from one track to an other will cost 1 shunt move, if the route includes a saw move, a move where the direction of the train changes, the cost is 2 shunt moves. A move of multiple units in one composition is counted as one move.
- Arrival break: Shunting units of the same arriving composition to different shunt tracks will cost 1 arrival break for each unit.
- Departure break: Selecting units for the same departing composition from different shunt tracks will cost 1 departure break for each unit.
- Not on preferred track: Each unit that is not parked on its preferred track.
- Not on empty track: Each unit that is not parked on an empty track.
- Wait time: Each minute a train unit is waiting on a platform before it is shunted or is departing (except for through trains).

- Parked on platform with departure: Each time a train unit is parked on a platform track on which an other train is departing.

In section 4.2 we have introduced the genetic algorithm that will search for good and feasible shunt schedules. However, this method will also generate infeasible solutions that will have to be evaluated by the simulation model. If the simulation model encounters an illegal move (e.g. capacity at a track is exceeded, other units are blocking a route) during the simulation, it will execute the move anyway such that the simulation can continue. However each violation will be recorded and at the end of the simulation processed into the fitness function as penalties. The following constraints can be violated during the simulation:

- Capacity: if a train unit is parked on a shunt or platform track that has not enough capacity left to store this unit.
- Crossing: when the route for the shunt activity is blocked by other units.
- Infra activity restriction: when a train arrives on a track where  $x$  minutes earlier a train has departed ( $x$  is location specific).
- Wrong departure order: if the composition of a departing train is in the wrong order.

### 4.4.3 Events

The events represent the processes that take place. We have specified the following events that occur during the shunting process:

- Arriving of a train at a platform: when this event occurs, a train composition enters the platform track from the A or B side of the station and can consist of multiple train units.
- Shunting from platform to shunt track: a train that arrives at a platform track can have one or multiple units that go out of service. When this event occurs, these units are shunted to their assigned shunt track.
- Shunting from shunt track to platform: a departing train that starts its service or needs extra units in its composition will have to get parked train units from the shunt yard. When this events occurs, the appropriate units are selected and shunted to the departing platform. Note that this event has to occur before the departing train leaves the platform.
- Departure of a train from a platform: when this event occurs, a train leaves the station from its scheduled platform, all the units that are needed for this departing train should already be parked on the platform track.

To give a brief description of the working of the simulation model we will walk through a few of the steps that occur during the simulation of a small timetable with a shunt schedule. In table 4.6 this timetable is presented with a shunt schedule provided by the GA. At the start of the simulation all the events are scheduled on the event list of the simulation model. For each arriving train an arrival event is scheduled and for each departing train a departing

Table 4.6: The timetable for the simulation example with a shunt schedule provided by the GA

Train nr	Arrival/Departure	Time	Platform	Units
1001	Arrival	10:00	p1	AB
1002	Arrival	10:20	p1	BD
2001	Departure	11:05	p1	BBA
2002	Departure	11:25	p1	D

Arriving unit nr:	1	2	3	4	1	2	3	4	Departure number
Parking gene:	1	1	1	2	3	2	1	4	Matching gene
Wait time	10	10	10	15	5	5	5	10	Early time gene

event is scheduled. For each unit that will be shunted to a shunt track, a shunt to shunt track (ShuntTST) event is scheduled. If multiple units are scheduled to be shunted at the same time from the same platform track, only 1 ShuntTST event is scheduled. In this example, the first 2 arriving units are shunted at the same time to the same shunt track. For each departing unit that has to be shunted from a shunt track to its departure platform, a shunt to platform track (ShuntTPT) event is scheduled. Once again, for units that are scheduled to be shunted at the same time to the same platform track, only 1 ShuntTPT is scheduled. In this example this holds for the first three departing units. Figure 4.7 gives a graphical representation of the scheduled events on the event list.

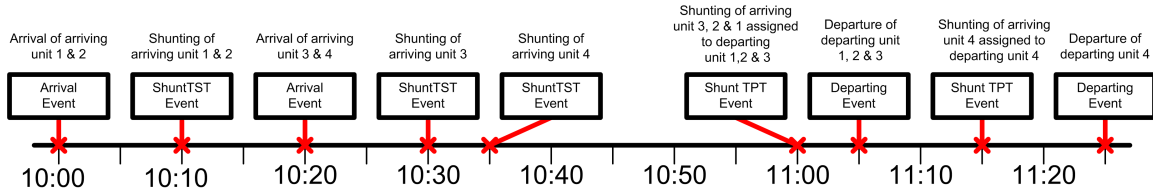


Figure 4.7: The events scheduled on the event list of the simulation

Once all the event are scheduled, the simulation starts by executing the events in order of appearance on the event list. First the arrival event of train number 1001 will be executed. This event puts all the train units belonging to this train on the platform track specified in the timetable. In case the track capacity is exceeded on the platform by this arrival, for each unit that exceeds the capacity a capacity penalty is recorded.

The second event on the list is a ShuntTST event. This event will start with selecting the units on the platform track that will need to be shunted at this point in time (simulation time). The selected units are shunted to their shunt tracks, specified by the shunt schedule. In this case both units are shunted to the same shunt track, so only one shunt move is recorded. If the units had different shunt tracks, 2 shunt moves were recorded and if both units are from the same arriving train an arrival break is also recorded. If any other unit is blocking the path to the shunt track, a crossing penalty is recorded for each shunted unit. The units are parked on their shunt track, and if any unit exceeds the capacity on his shunt track, a capacity penalty is recorded. The following arrival event and the two ShuntTST



events will follow this same procedure.

The fifth event on the list is a ShuntTPT event. Depending on the optimization approach that is used, two things can happen. In the case of the integral approach, the shunt schedule dictates which train unit on the shunt yard will be used for the departing train unit. In this example, 3 departing units are needed, and the shunt schedule says that arriving unit 1, 2 and 3 are used. All three units are parked on the same shunt track, and are already parked in the correct departing order. This means that only 1 shunt move is recorded. In the case of the rule based approach, the shunt schedule provides this matching of departing units to arriving units. In this case, the departure rule described in section 4.3.2 will assign three units from the shunt yard to the departing units. Shunt moves, departure breaks, crossing penalties and capacity penalties are recorded the same way as with the ShuntTST event.

The next event on the event list is a departing event. This event checks if all the departing units on the departing platform are in the correct order. The units will leave the station and if an other unit is blocking the path, a crossing penalty is recorded. The next ShuntTPT and departing event will follow the same procedure. Once all the events have been executed, the fitness is calculated by multiplying all the recorded performance statistics with their corresponding weights.

During a ShuntTPT event, two unique situations can occur because of the flexible shunt times. In the first situation a matched unit is still waiting for his ShuntTST event on its arriving platform track at the moment it is needed for a departure. In this case the ShuntTPT event will cancel the scheduled ShuntTST event. If this unit is not waiting on the departing platform track, it will shunt the unit to its correct platform track, recording the shunt moves and possible crossing penalties. In the second situation a departure unit is assigned to a unit that has not yet arrived at the station. This can happen when the shunt time is chosen too far in advance of the actual departure. In this case, a new ShuntTPT event is scheduled just before the departing time of the unit.

#### 4.4.4 Verification of the simulation model

Model verification can be defined as “ensuring that the computer program of the computerized model and its implementation are correct” [29]. Without verification, one can never tell if the model is computing the correct results and without correct results we can never evaluate different shunt schedules. In our case we want to make sure that our simulation model is correctly implemented and that it computes the right results given a shunt schedule. A few test cases have been developed to see whether the simulation behaves like it is supposed to. We tested whether all the events occurred on the right time, whether the performance statistics were correctly recorded and whether the components behaved as they are supposed to. For this purpose a visualisation of the entire shunting process was developed that visualises each state change in the system. Figure 4.8 shows this visualisation. On the blue panel you see the performance statistics that have been recorded until this moment in the simulation. On the left side of the grey panel, the shunt tracks are drawn. The colored circles represent train units that are currently using the tracks, the different colors represent different subtypes. On the right side of each track the currently used capacity

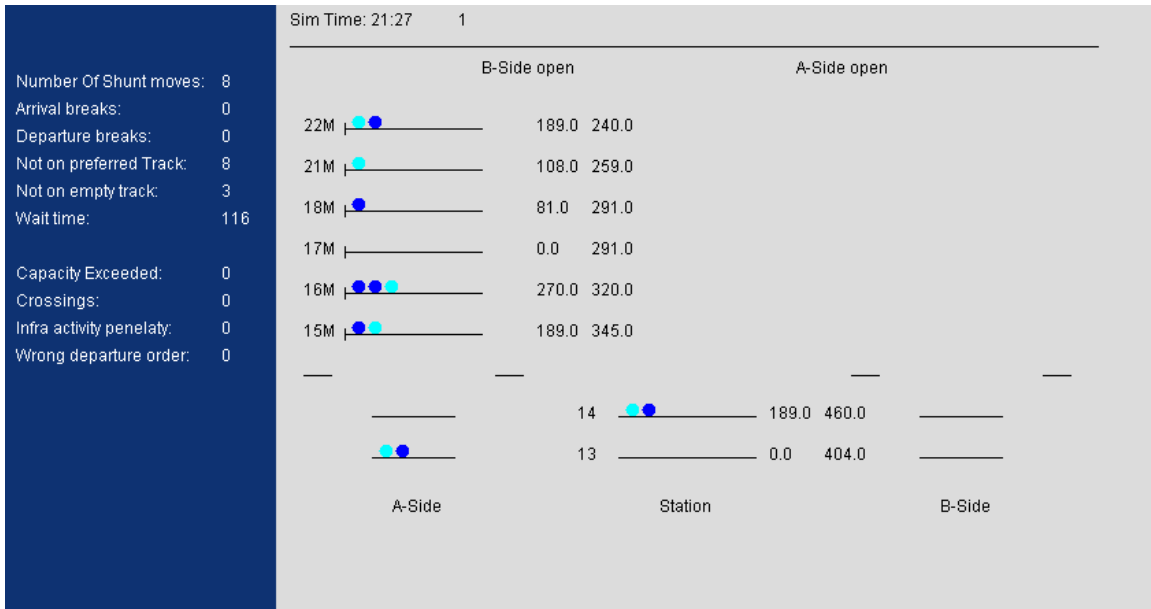


Figure 4.8: Visualization of the shunting process

and the total length of the track is shown. This station has two platform tracks, on one track a train is waiting and from the A side of the station a train with two train units is arriving at the station. By walking step by step through the simulation, we can see what happens with each component and the statistics each time the state of the system changes. Based on these evaluations and by comparing the results of a shunt schedule created by the MIP approach with the result recorded by our simulation with this same schedule, we can verify that the simulation model is simulating the shunt process correctly and is computing the result correctly.

## Chapter 5

# Experiments: Theoretical cases

In this chapter we evaluate the two alternatives approaches described in the previous chapter. Both approaches will be tested with the use of four theoretical cases. Based on the results we will select the best performing approach that will be used for the real-world experiments. In the remaining of this chapter we will refer to the normal approach as the *integral approach* (IA) and the alternative approach as the *rule based approach* (RBA)

### 5.1 The test cases

Four test cases have been developed for the evaluation of our approaches. The cases describe a theoretical shunt problem, including a station, shunt yard and a timetable. In the first three cases we did not include variable shunting times. This way we can compare our results with the MIP approach that assumes that shunt times are fixed. The fourth case describes a problem where changing some of the shunt times will improve the shunt solution.

The first case is a small shunting problem for which we can compute the optimal solution by evaluating every possible solution in the search space. With this case we can see if and how fast our approaches can reach this optimum. The second case is a larger problem for which the number of solutions in the search space is so big that it is impossible to evaluate each solution. However the case is created in such a way that we can find the optimum by hand. The third case is much more complex and the optimum is not known to us.

#### Case 1

In this test case 6 train compositions arrive during the day and 5 depart. The compositions consist of a total of 12 units, from a range of 3 different subtypes (A,B and C). There is one arrival and departure platform and 3 different shunt tracks, which are all LIFO tracks. Figure 5.1 illustrates this shunting location. In table 5.1 you find the timetable of all arriving and departing trains. Because this is quite a small case we can find the optimum by evaluating each possible solution. For each arriving train unit we can choose from 3 different shunt tracks, so in total there are  $3^{12}$  different parking solutions. In this case there are 768 different matching solutions possible. This means that in our integral approach we have a total of 408,146,688 different solutions (parking solutions  $\times$  matching solutions). The rule based approach has 531,411 different solutions (only the parking solutions). Each evaluation

Table 5.1: Timetable for case 1

Time	1	2	3	4	5	6	7	8	9	10	11
Arriving Trains	BA	B	CA		AA		CBA	BC			
Departing Trains				BB		AC			ABA	CC	AAB
Platform	1	1	1	1	1	1	1	1	1	1	1
Side	A	A	A	A	A	A	A	A	A	A	A

by the simulation model for this case takes about 0.00008 seconds. Which means that we can evaluate each solution in the integral approach in about 9 hours and each solution in the rule based approach in about 43 seconds.

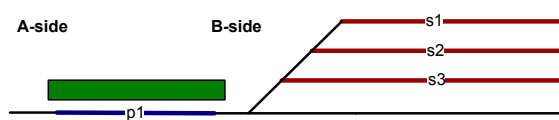


Figure 5.1: Layout of the shunting location in case 1

## Case 2

In the second case, each train that arrives has only 1 unit. First 5 trains arrive of subtype A, followed by 5 trains of subtype B, 5 of subtype C, 5 of subtype D and 5 of subtype E. After these 25 trains have arrived, 5 trains each with 5 units have to depart from the station, in the following composition: EDCBA. The complete timetable can be found in Appendix B.1. There are 5 shunt tracks at the shunt yard. The optimal solution would be to park each A unit at a different shunt track, each B unit at a different shunt end so on (see figure 5.2). This way, each departing train is already parked in the correct composition at the shunt yard, minimizing the total shunting effort. This seems a quite trivial problem, however in total there are  $5^{25}$  different solutions, for each arriving unit we can choose between 5 different shunt tracks. To make the search space even bigger, we repeat this arrival and departure sequence 4 times making the search space  $5^{100}$ . Today's computers are not able to evaluate that much solutions, however we know the optimum solution, so we can examine how close our approaches reach this optimum and how fast.

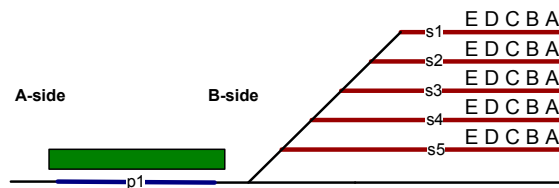


Figure 5.2: Optimal parking layout after the first 25 trains have arrived

### Case 3

In the third case we have randomly generated a timetable with 100 arriving units and 100 departing units in trains with composition size between 1 and 3 units. There are 6 different train subtypes and 15 shunt tracks. In total there are 48 arriving trains and 49 departing trains. All the arrivals occur before the first departure, which means that we have to park 100 units at the same time at the shunt yard. This makes the problem very hard to solve. There are numerous ways to park the arriving units, but we want to park these units in such a way that we minimize the shunting effort. In this case we don't know the optimal solution, but we can examine how well the different approaches can handle such a case. The complete timetable can be found in Appendix B.2.

### Case 4

This fourth case describes a small shunting problem, with 2 shunt tracks and 2 platform tracks. A total of 20 train units arrive and depart at the station. The timetable is created in such a way that changing the shunt time for a few units will decrease the shunting effort. The search space includes for each arriving unit 10 different waiting times and for each departing unit 10 different early times. The model will try to find good shunt times such that the shunting effort is minimized. Table 5.2 shows this timetable. The highlighted units are the units that can benefit from changing their shunt time.

Table 5.2: Timetable for case 4

Time	10:00	10:10	10:20	10:30	10:40	10:50	11:00	11:10	11:20	11:30
Arrivals	AB	AC			CC	A A			AB	B D
Departures			BA	CA			AC	AC		
Platform	1	1	2	2	1	2	1	2	1	1
Side	A	A	A	A	A	A	A	A	A	A
Time	11:40	11:50	12:00	12:10	12:20	12:30	12:40	12:50	13:00	13:10
Arrivals		AA		AA	B		CA	A		
Departures	BBA		DAA			BAA			AA	C
Platform	2	2	2	1	1	2	2	1	1	1
Side	A	A	A	A	A	A	A	A	A	A

In figure 5.3 we show the track occupation diagram for this example, using the most beneficial shunt times. In this figure you can see a timeline spanning the entire timetable. Each line represents a track and each block represents a train unit. Units closest to the track line are parked closest to the b-side of that track. The length of a block shows the time this unit is parked on the track. We categorise 4 different types of “benefits” that can be obtained when shifting the shunt time. First a unit can wait on the platform until it has to depart, this way we do not have to shunt this unit and we save 2 shunt moves. In this example the second A unit that arrives at 10:50 is using this strategy. The second type is when a unit is waiting for other arriving units with which it can shunt together to a shunt track, the A and B unit arriving at 11:20 are such examples, they wait until the next train arrives and shunt together with the B unit of that train to shunt track 1. This

way one shunt move is saved. The third group are units that are shunted to their departure platform together with earlier departing units to save a shunt move. Unit C departing at 13:20 is an example of this type. The last type consists of units that are waiting or are shunted earlier to their platform to create a better parking order. A train unit of type D arriving at 11:30 is an example of this. This unit waits until the two A type units, that arrive 10 minutes later at the station, are shunted to a shunt track. Hereafter unit D is shunted to the same shunt track to form the departing composition that is needed at 12:00. With this timetable we can save up to 6 shunt moves when applying flexible shunt times. To make it a bit more difficult for the genetic algorithm, the complete timetable we test in this case will repeat this sequence 2 times. Which means that in total there are now 40 arriving units and 40 departing units, each with 10 shunt time possibilities.

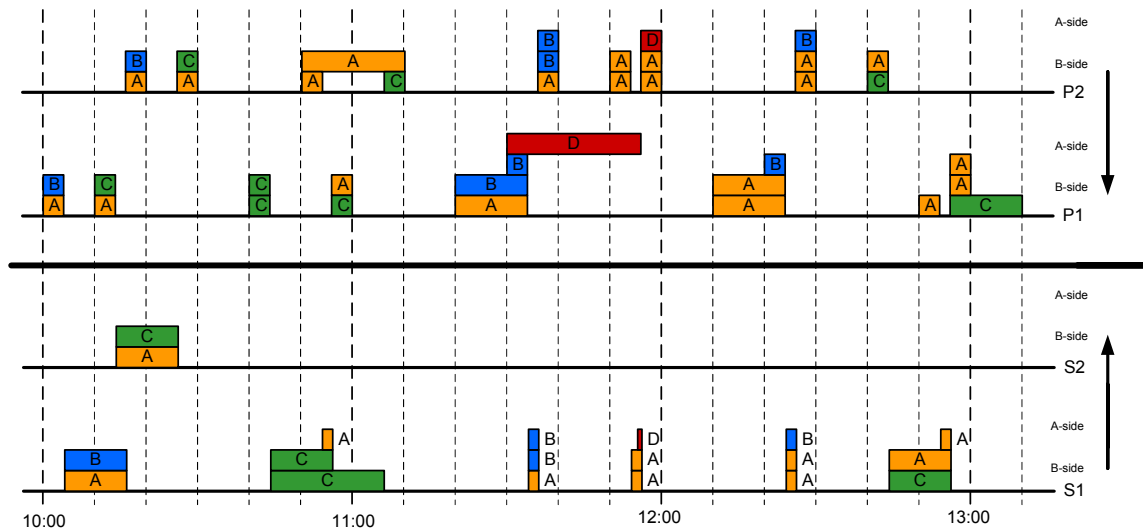


Figure 5.3: Track occupation diagram

## 5.2 Experiments

As we have explained in section 4.2, the user should specify the settings of the GA. Before we started these experiments we tested each case with different settings and used the most suitable ones for each case. In smaller cases we can use less generations and a smaller population size because the search space is much smaller. To show the impact of using larger population sizes and more generations on the computation time and the results, we included results of two different settings. In the first setting we tried to minimize the number of evaluations (generations \* population size) for which we still get acceptable results and in the second setting we tried to maximize the results within an acceptable computation time (15 minutes). Table 5.3 shows these settings. The experiments were performed on a PC with an Intel Xeon CPU at 3 GHz with 3.25 GB of RAM operating under Windows XP. For each experiment we use the same initial population. The individuals in both approaches will start with exactly the same parking genes, they only differ from each other on the fact that the individuals in the rule based approach do not have the matching gene. The search

will terminate once all the generations have been completed.

Table 5.3: Settings for theoretical cases

Case 1			Case 2		
Objectives	Weights		Objectives	Weights	
Shunt moves	1		Shunt moves	1	
Arrival breaks	2		Arrival breaks	0	
Departure breaks	5		Departure breaks	5	
Capacity penalty	10		Capacity penalty	10	
Crossing penalty	10		Crossing penalty	10	
Settings	Exp 1a	Exp 1b		Exp 2a	Exp 2b
Generations	20	100	Generations	100	1000
Population size	20	100	Population size	100	200
Crossover rate	0.9	0.9	Crossover rate	0.9	0.9
Mutation rate	0.9	0.9	Mutation rate	0.9	0.9
Elitism	2	5	Elitism	5	10

Case 3			Case 4		
Objectives	Weights		Objectives	Weights	
Shunt moves	1		Shunt moves	1	
Arrival breaks	2		Arrival breaks	0	
Departure breaks	5		Departure breaks	5	
Capacity penalty	10		Capacity penalty	10	
Crossing penalty	10		Crossing penalty	10	
Settings	Exp 3a	Exp 3b		Exp 4a	Exp 4b
Generations	100	1000	Generations	200	600
Population size	100	500	Population size	100	400
Crossover rate	0.9	0.9	Crossover rate	0.9	0.9
Mutation rate	0.9	0.9	Mutation rate	0.9	0.9
Elitism	5	10	Elitism	5	10

### 5.3 Results

Because genetic algorithms have random behaviour, they can produce different solutions each time you run the experiment. Therefore we repeated each test 10 times and show you the best, worst and average result.

**Results case 1** Table 5.4 shows the results of the experiments for Case 1. The MIP model was able to find the optimal solutions within a second. Both approaches were also able to find the optimal solution 1 of the 10 times with the small GA settings. And they complete their search in 0.03 seconds. The integral approach did only find 3 out of the 10 times a feasible solution and the average fitness is almost twice the fitness of the optimal solution. The rule based approach on the other hand found 10 out of the 10 times a feasible

solution and its average fitness lies close to the optimal fitness. In the “heavier” GA setting the rule based approach found 9 times the optimal solution, the integral approach only 5 times and even computed 1 infeasible solution. The computation time however was a bit lower than the rule based approach. Based on these results we can conclude that both approaches are quite able to find optimal solutions, however the rule based approach was significantly better than the integral approach. However, if we make the population size even larger we see that both approaches always find the optimal solutions within less than 2 seconds.

**Results case 2** Table 5.5 shows the results for the experiments with case 2. Once again the integrated approach has a lot more trouble finding feasible solutions than the rule based approach. Both approaches were able to find the optimal solutions, the rule based approach found the optimum every time when we used the heavier settings. Also the MIP approach did not have much trouble finding the optimal solution, in less than a minute this optimum was found. In figure 5.4 shows in how many generations both our approaches found the optimum solution. The rule based approach does not only finds the optimum more often, but also needs less evaluations to reach this optimum.

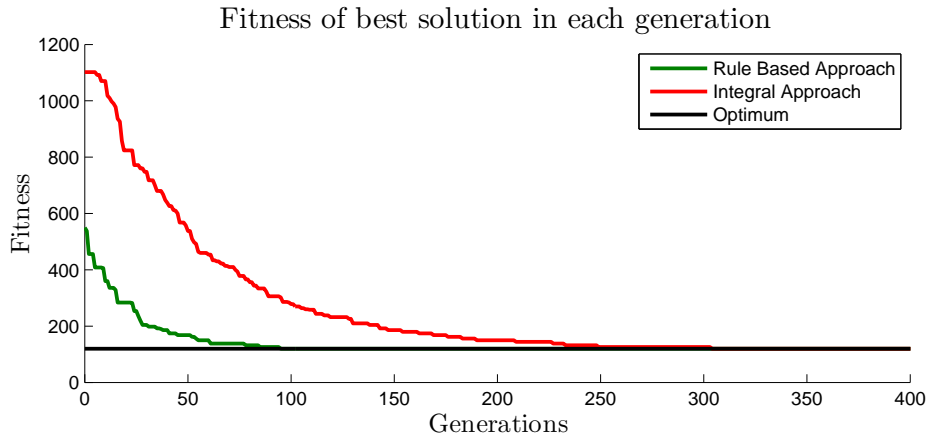


Figure 5.4: Performance of the best solution of each generation

**Results case 3** Results for Case 3 can be found in table 5.6. This time we can not compare our results with the MIP model because MIP solver was not able to build the model within a reasonable amount of time. As we have mentioned earlier in this section, case 3 is a very complex case. Because all the departing trains depart after the last arriving train, a lot of potential crossings exist. In case 3, in theory, each unit has 16 possible matches (100 units / 6 different types), and each match can on average be crossed by half of the other matches on 15 different tracks (the first departing unit by all, the last departing unit by none). In total there are  $16 * 100 * (16 * 100 / 2) * 15 = 19,200,000$  crossing constraints the model has to construct before it can even start the search. After 16 hours no solution was found and we stopped the search.

For experiment 3a we see that the integral approach performs a lot worse than the rule based approach. Also in experiment 3b the rule based approach achieves better results,



Table 5.4: Results for theoretical case 1

	over 10 runs	Shunt moves	Arrival breaks	Departure breaks	Capacity penalties	Crossing penalties	Total fitness	Computation time in seconds	Optimal	Feasible
Optimum ia	-	16	3	2	0	0	32	32650	-	-
Optimum rba	-	16	3	2	0	0	32	43	-	-
MIP approach	-	16	3	2	0	0	32	<1	-	-
Expla ia	best solution	16	3	2	0	0	32	0.03		
	worst solution	20	5	4	0	1	60	0.03		
	average	18.2	3.7	3.5	0	0.8	51.1	0.03	1	3
Expla rba	best solution	16	3	2	0	0	32	0.03		
	worst solution	19	5	3	0	0	44	0.03		
	average	17.3	3.6	2.7	0	0	38	0.03	1	10
Explb ia	best solution	16	3	2	0	0	32	0.62		
	worst solution	16	2	4	0	0	41	0.62		
	average	16.2	3.2	2.1	0	0.1	34.2	0.63	5	9
Explb rba	best solution	16	3	2	0	0	32	0.88		
	worst solution	17	4	2	0	0	35	0.86		
	average	16.1	3.1	2	0	0	32.3	0.89	9	10

Table 5.5: Results for theoretical case 2

	over 10 runs	Shunt moves	Arrival breaks	Departure breaks	Capacity penalties	Crossing penalties	Total fitness	Computation time in seconds	Optimal	Feasible
Optimum	-	120	0	0	0	0	120	-	-	-
MIP approach	-	120	0	0	0	0	120	46	-	-
Exp2a ia	best solution	144	0	24	0	10	364	5.6		
	worst solution	147	0	27	0	14	422	5.3		
	average	145.6	0	25.6	0	11.7	390.6	5.4	0	0
Exp2a rba	best solution	124	0	4	0	0	144	8.5		
	worst solution	126	0	6	0	2	176	8.5		
	average	124.9	0	4.9	0	0.5	154.4	8.52	0	6
Exp2b ia	best solution	120	0	0	0	0	120	134.9		
	worst solution	127	0	7	0	1	172	135.4		
	average	123.2	0	3.2	0	0.4	143.2	135.6	1	7
Exp1b rba	best solution	120	0	0	0	0	120	208.4		
	worst solution	120	0	0	0	0	120	210.3		
	average	120	0	0	0	0	120	209.1	10	10

Table 5.6: Results for theoretical case 3

MIP approach	over 10 runs	Shunt moves	Arrival breaks	Departure breaks	Capacity penalties	Crossing penalties	Total fitness	Computation time in seconds	Feasible
Exp2a ia	best solution	184	43	44	0	13	620	5.8	
	worst solution	189	46	46	1	29	811	5.9	
	average	185	44.6	43.4	0.2	19.2	685.2	5.8	0
Exp2a rba	best solution	165	43	25	0	0	376	13.7	
	worst solution	168	43	28	0	2	414	13.9	
	average	168.3	44.1	27.2	0	0.3	396.5	13.7	8
Exp2b ia	best solution	153	41	15	0	0	310	308.3	
	worst solution	161	38	26	0	1	377	313.2	
	average	157	38	22	0	0.3	346	313.3	8
Exp2b rbh	best solution	138	31	10	0	0	250	714.1	
	worst solution	149	35	17	0	0	304	718.4	
	average	144	32.9	14.1	0	0	280.3	724.4	10

although the differences are a bit smaller. The huge differences in results between the two approaches in the short experiment (3a) is mostly due to the number of possible crossings. The rule based approach will always try to minimize the crossings by selecting departing units that are in front of the shunt track. The integral approach does not have any information on possible crossings when making the matching gene. Therefore it takes a lot more time to find even a feasible solution. Figure 5.5 illustrates this difference between both approaches. The computation time of the rule based method is more than twice that of the integral approach. The decision rule evaluates each parked unit every time a departing unit needs to be shunted. Because there are so many units parked at the shunt yard this takes a bit more time.

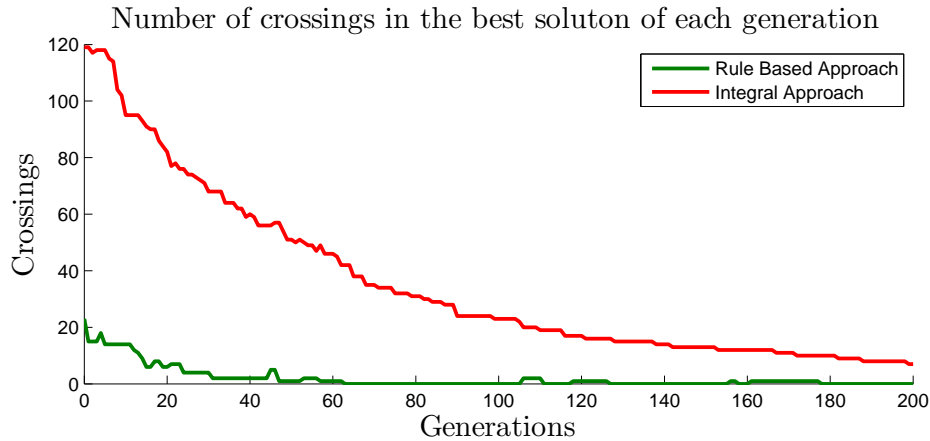


Figure 5.5: Performance of the best solution of each generation

**Results case 4** Results for case 4 are shown in table 5.7. If we exclude the flexible shunt times, we see that our approach has no trouble finding the optimal solution. Also the MIP approach finds this optimum within 4 seconds. However if we introduce the flexible shunting times, we see that it takes a lot more time to find the optimum. This was expected since we have increased the search space with a factor  $10^{40}$  (10 shunt times for each arriving and departing unit). With the “heavier” settings both approaches were able to find this optimum. In contrast with the other cases, this time the integral approach performed slightly better than the rule based approach. In this case there are only 128 different matching possibilities, thus finding a good matching solution is not that hard. The integral approach is not bound by some rule, it can search over each possible solution and therefore in this case, since the matching itself is not a problem, it might have more directions for finding the optimal solution than the rule based approach.

### Extra experiment

One thing that one might have noticed is that the genetic algorithm, especially in combination with the rule based decision, is capable of finding acceptable solutions very fast, but finding near optimal solutions can take a lot of time. To reduce this time, we tried to use the solutions found with the “fast settings” as initial population for the “heavier settings”.

Table 5.7: Results for theoretical case 4

	over 10 runs	Shunt moves	Arrival breaks	Departure breaks	Capacity penalties	Crossing penalties	Total fitness	Computation time in seconds	Optimal	Feasible
RBA without time	-	48	6	2	0	0	58	3.5	-	-
MIP approach	-	48	6	2	0	0	58	3.6	-	-
Optimal with time	-	36	0	0	0	0	36	-	-	-
Exp4a IA	best solution	39	2	2	0	0	49	4.3	0	9
	worst solution	46	4	2	0	0	71	4.2	0	9
	average	44.4	4.3	2.8	0	0.1	59.4	4.2	0	9
Exp4a RBD	best solution	39	1	1	0	0	44	6.7	0	10
	worst solution	41	2	2	0	0	51	6.6	0	10
	average	39.9	1.6	1.8	0	0	48.9	6.3	0	10
Exp4b IA	best solution	36	0	0	0	0	36	47.9	5	10
	worst solution	37	1	1	0	0	42	48.6	5	10
	average	36.9	0.8	0.1	0	0	37.4	49.1	5	10
Exp4b RBA	best solution	36	0	0	0	0	36	73.4	4	10
	worst solution	38	2	2	0	0	48	79.1	4	10
	average	36.6	0.3	0.5	0	0	39.1	76.7	4	10

This way we might be able to reduce the computation time, since the “low settings” are able to converge to an acceptable solution much faster, after which the “heavier settings” can take over to find near optimal solutions. Moreover if we use a couple of different initial solutions it could result in better end results. In this experiment we are going to use 1, 5 and 10 initial solutions found with the settings of Exp3a. The remaining part of the initial population will be created randomly. Figure 5.6 shows the change in fitness of the best solution against the computation time. Including good initial solutions does indeed reduce the computation time for finding acceptable solutions, however it does not reduce the computation time of finding the same quality solutions that were found with the “heavier setting” without these initial solutions. The graph shows that the experiment with 1 initial solution found the best solution, however the difference with the other three experiments is not significantly and in different test runs it did not always find better solutions than the experiment without initial solutions.

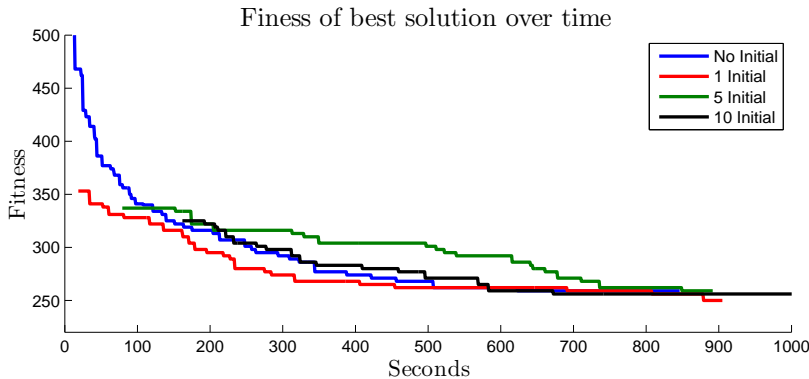


Figure 5.6: Performance with initial population found by the GA

## 5.4 Conclusion

Based on the experiments in this chapter we can conclude that both our approaches are capable of solving the train unit shunting problem quite well. Compared with the MIP approach the results for case 1 and 2 are well matched. Although our approach not always computes the optimum result. This however can be overcome by increasing the population size and the number of generations or by repeating the same experiments a few times. In this situation the MIP approach computes the optimum much faster. In case 3 and 4 we see that our approaches are superior to the MIP approach. In case 3 the MIP approach was not even able to find a feasible solution within half a day, while our approaches were able to find quite good solutions in less than an hour. The second requirement described in section 3.3 which stated that we want to be able to solve larger problems is thereby achieved. In case 4 we see that our approaches were able to handle flexible shunt times, and thereby finding better shunt schedules. The current MIP approach does not have this flexibility and is not likely to ever introduce this flexibility due to the problem of handling the enormous increase of decision variables. Our first requirement of introducing flexible shunt times was successfully implemented in our approach.

If we compare our two approaches with each other, we see that in most cases the rule based approach performs a lot better than the integral approach. We already stated that this is probably due to the fact that the integral approach is searching good matchings from scratch without knowledge of the problem, while the rule based approach selects the best matches for a given parking solution. This problem could not be solved by providing the integral approach more time for finding its solution by increasing the population size and the number of generations. In case 3 we saw that increasing the settings of experiment 3b only lead to longer computation times, without any improvements on the results. The integral approach tends to get stuck at local optima much faster than the rule based approach. Our explanation for this outcome is that the matching and parking variables are so much correlated that changing one type of variable while keeping the others the same almost always leads to worse solutions once a local optimum has been found. In the rule based approach we only change the parking variables and for each new solution suitable matches are found by the rule. The downside of this rule based approach is that we can not find each possible solution. Our rule always selects the best match for a departing unit at the moment this unit is departing. However this best match does not take future departures into account, so it could happen that the “best” decision in the present can lead to much worse situation in the future. Of course such a solution would get a low evaluation, and the GA should be able to find different parking solutions such that this earlier departure unit is forced to choose an other match by giving him a better alternative or by blocking the best match with an other unit. In our research we did not find any obvious problems encountered by the rule. However it could be possible that in some cases the optimal solution can never be found using such a rule.

Based on the results we can conclude that overall the rule based approach is performing a lot better than the integral approach. Therefore in the remainder of this research we are using this approach. In this chapter we evaluated 4 different cases, all constructed by hand and describing some theoretical situations. Based on the results we concluded that our approaches are performing very well. However our research objective is to develop a method that can solve real world shunting problems. In the next chapter we will challenge our rule based approach with two real world cases in the Netherlands.





## Chapter 6

# Experiments: Real-world cases

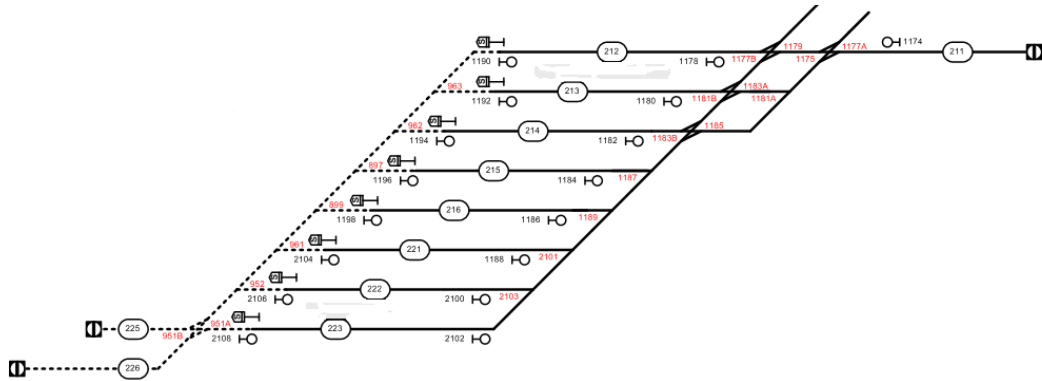
In this chapter we discuss the experiments and their results on two cases taken from real-world examples. Both cases describe an existing shunt location in the Netherlands and data from the NS timetable of 2008 were used as input for the shunt problem. We compare the results with the actual planned shunt schedules created by the planners at NS. We start this chapter with a section that describes the shunting problem at Hoofddorp, the experiments we have performed and the results that have been found. Thereafter the same procedure is followed for the shunting problem at Rotterdam. We conclude this chapter with a discussion on the applicability of our approach as a planning tool for local shunt planners.

### 6.1 Hoofddorp

The real world case used in this section is the shunt yard near a town called Hoofddorp (Hfdo) in the Netherlands. This shunt yard stores trains for which the last stop is either at station Schiphol (Amsterdam Airport) or at station Hoofddorp. The shunt yard is located outside the Hoofddorp station area which means that not only trains that stop their service at Schiphol but also at Hoofddorp will have to travel empty over the main railroad network to arrive at the shunt yard. In the Netherlands, each company using the main railroad network has to reserve track capacity for each train. Therefore these empty trains are scheduled as new train services from Schiphol or Hoofddorp to the shunt yard and the other way around. This means the arrival and departure times at the shunt yard are scheduled in advance (i.e. are present in the timetable) and therefore the shunting times are dictated by the timetable. Therefore no decision has to be made on the shunting times. The layout of the shunt yard is shown in figure 6.1. Although the shunt tracks can be accessed from both sides (free tracks), this functionality is never used in shunt schedules for Hoofddorp. Therefore these shunt tracks can be treated as LIFO shunt tracks.

The train units in the timetable that have to be parked at shunt yard Hoofddorp are divided into 10 different subtypes, table 6.1 shows these subtypes and their lengths.







**Hfdo preferences and special restrictions** Each shunt yard location has its own preferences and network restrictions. This means that for each location, different weights



Shunt track	212	213	214	215	216	221	222	223
Length in meters	441	362	366	369	369	411	377	432

Figure 6.1: Shunt yard near Hoofddorp

Table 6.1: Train units at shunt yard Hoofddorp

Type	Subtype	Length in meters	
ICM	ICM-3	81	
	ICM-4	108	
IRM	VIRM-4	109	
	VIRM-6	162	
DMM	MDDM-4	102	
	DD-AR-4	124	
Mat'64	Mat'64-2	53	
	Mat'64-4	102	
SPR	SGMM-3	79	
International	K-NDC	243	

should be applied for the objective functions and sometimes additional restrictions have to be added to the model. At the shunt yard in Hoofddorp planners have preferred tracks for different train series which can be found in table 6.2. The train series numbers correspond with the last 4 digits of the train numbers in the timetable (see Appendix B.4 for a timetable of Hoofddorp). Parking trains on empty shunt tracks is preferred by the shunting crew because this speeds up the shunting process. Entering a track with another train parked on it means that a lower speed limit has to be applied for safety reasons. This empty track preference is considered more important by the planners of Hoofddorp than parking on a preferred track.

Every 2 hours an international train from Germany to Schiphol is parked at the shunt yard. This train is different from the normal trains used in the Netherlands; it consists of a locomotive with carriages for which the locomotive has to be in front of the train. This means that, when the train is parked at the shunt yard, the locomotive has to be shunted

Table 6.2: Train Series and their preferred track

Tracks	212	213	214	215	216	221	222	223
Train Series	3900	4300	3300	700	3500	3100	100, 200	100, 200
				1600			1100, 1200	1100, 1200

to the other side of the train before the train can leave the shunt yard. Planners have assigned three tracks on which this international train should be parked, namely track 223, 222 and 221. For the time this train is parked, no other train can be parked at this track and there has to be a time window of at least 10 minutes in which the locomotive can use track 221, 222 or 223 to get in front of the train, meaning that one of these tracks has to be empty for at least 10 minutes during its stay. These requirements are treated as restrictions meaning that no other tracks can be used for parking the international train and shunting the locomotive.

Finally the infrastructure of the shunt yard causes an extra restriction. In the Netherlands there is an infrastructure restriction saying that there has to be at least 4 minutes between two trains that use the same track. This means that we cannot enter a shunt track on which a train departed within these 4 minutes. In Hoofddorp the infrastructure is even a bit more restricted. If you take a better look at figure 6.1, you can see that in Hoofddorp, arrival and departure routes to and from most shunt tracks use the same track. Therefore a departure from any track means that no train can arrive at track 214 to track 223 within 4 minutes. If the departure is at track 214 to track 223, only track 212 and 213 are accessible within these 4 minutes, during a departure at track 213 only track 212 can be accessed and a departure from track 212 means that no track is accessible.

### 6.1.1 Experiments

In the first experiment we are going to solve a shunt problem for one day, hereafter we introduce some new features that can help solving the problem and in the final experiment we are trying to solve a shunt problem for the entire weekend.

**Experiment 1** In the first experiments we are trying to generate a shunting schedule for a Tuesday evening to Wednesday morning in October. This time period is chosen because it illustrated the ending and start-up of a normal day, the period where decisions have the greatest impact on the performance of the plan because the shunt yard’s capacity is used the most in the night; throughout the day most trains are in service. During the time interval from Tuesday 18:00 to Wednesdays 10:00, 96 trains arrive at the shunt yard with a total of 118 train units. These units depart in 95 different train compositions. Table 6.3 shows the settings for this experiment. We selected these settings after trying a lot of different settings. Increasing the number of generations or the population size did not result into better solutions, in most cases the best solution was already found after 300 generations. The weights of the objectives are chosen quite close to that of the restrictions. If we would penalise the restrictions too hard in comparison with the objective, we see that the algorithm often gets stuck in local optimums. One thing that is remarkable in our settings is the proportion of the population size and the number of generations. It is widely

recognised that keeping the population size small while having a lot of generations normally results in better performance of the genetic algorithm. In our case, this is true if we want to realise an acceptable solution (i.e. feasible) in a small amount of time. However we found out that because of the highly constrained nature of our problem, a small population size most often leads to local optima. This could not be fixed by using more generations, but increasing the population size did have a positive effect on the results.

Table 6.3: Settings for Hoofddorp experiment 1 &2

Objective	$w$	Restrictions	$w$	GA parameters	
Shunt move	5	Capacity	50	Generations	500
Arrival break	20	Crossing	50	Population Size	400
Departure break	30	Infrastructure	50	Crossover Rate	0.9
Not on preferred track	1	International train	50	Mutation Rate	0.9
Not on empty track	10			Elitism	40

**Experiment 2** Although including initial solutions did not increase the performance of the GA for the theoretical cases, for this Hoofddorp case we will try a similar approach. Only this time we will not use initial solutions found by the GA, but a solution that includes some of the information we already have. This initial solution will park each train on its preferred track. Because these preferences are formulated by the planners based on their experience, this usually gives quite a good answer. The same settings of experiment 1 are used.

**Experiment 3** In this third experiment we are going to find a shunting solution for a complete weekend at Hoofddorp. This means that we have to find a suitable parking track for each train that arrives from Friday evening until Monday morning. In total 514 units arrive in 485 different compositions and depart in 478 different compositions during this interval. A weekend shunt plan is often a bit more complicated than the plans during the week. In the weekend not all trains are in service, causing some trains to be parked during the entire weekend at the shunt yard. Solving each day in the weekend separately could lead to bad or infeasible solutions. For the trains that have to be parked the entire weekend you will need to park that train each day on the same track. However the optimal parking track for such a train on the first day could perhaps not be used the second day. To find the optimal track for the entire weekend, you should solve each day of the weekend together in one go. For this experiment we will use almost the same settings as in the earlier Hoofddorp experiment. Only this time we have to use more generations and a larger population size because the number of decision variables is almost 5 times larger than in the first two experiments. The genetic algorithm needs a lot more time to find a good solution in this space. The weights of the penalties have been set a lot higher because the model has more trouble finding a feasible solution in this large search space. Table 6.4 shows the settings for experiment 3.

Table 6.4: Settings for Hoofddorp experiment 3

Objective	$w$	Restrictions	$w$	GA parameters	
Shunt move	5	Capacity	100	Generations	500
Arrival break	20	Crossing	100	Population Size	1000
Departure break	30	Infrastructure	100	Crossover Rate	0.9
Not on preferred track	1	International train	100	Mutation Rate	0.9
Not on empty track	10			Elitism	100

### 6.1.2 Results

Table 6.5 shows the results for experiment 1 & 2 of the Hoofddorp case. Once again we have executed 10 different runs for each experiment to obtain a best, worst and average result over these 10 runs. We compare the results with the results of the MIP approach. This result has already been verified by the actual planners of Hoofddorp to be a good shunt schedule.

Our approach was able to find feasible solutions in each run. The results for experiment 1 show that most objectives are optimized, only with the track preference objective our approach scores a lot worse than the MIP approach. This objective has the lowest weight in our fitness function, and therefore it is apparently a bit neglected in the evolution process. The introduction of the initial solution in experiment 2 seems to overcome this problem. In this case the evolution process starts with a solution where all the units are parked on their preferred track, and during the evolution process the other objectives are tried to be optimized. The results for these experiments even outperform those of the MIP approach in 3 of the 10 runs. Our approach also uses less than half of the computation time of the MIP approach.

Table 6.5: Result for Hoofddorp experiments 1 &amp; 2

	Experiment 1			Experiment 2			MIP
	best	worst	average	best	worst	average	
Shunt moves	192	193	192.5	192	193	192.2	192
Arrival breaks	1	1	1	1	1	1	1
Departure breaks	0	1	0.5	0	1	0.2	0
Not on preferred Track	61	72	67.4	23	30	28.5	34
Not on empty Track	7	9	7.6	7	9	7.8	7
Capacity penalties	0	0	0	0	0	0	0
Crossing penalties	0	0	0	0	0	0	0
Infra. penalties	0	0	0	0	0	0	0
Int. Train penalties	0	0	0	0	0	0	0
Total fitness	1111	1177	1140.9	1073	1135	1093.5	1084
Computation time in seconds	389	425	412	397	407	407	900

Figure 6.2 shows a shunt schedule computed by our model. In this figure each grey block represents a shunt track. The height of this block shows the length of the shunt track and

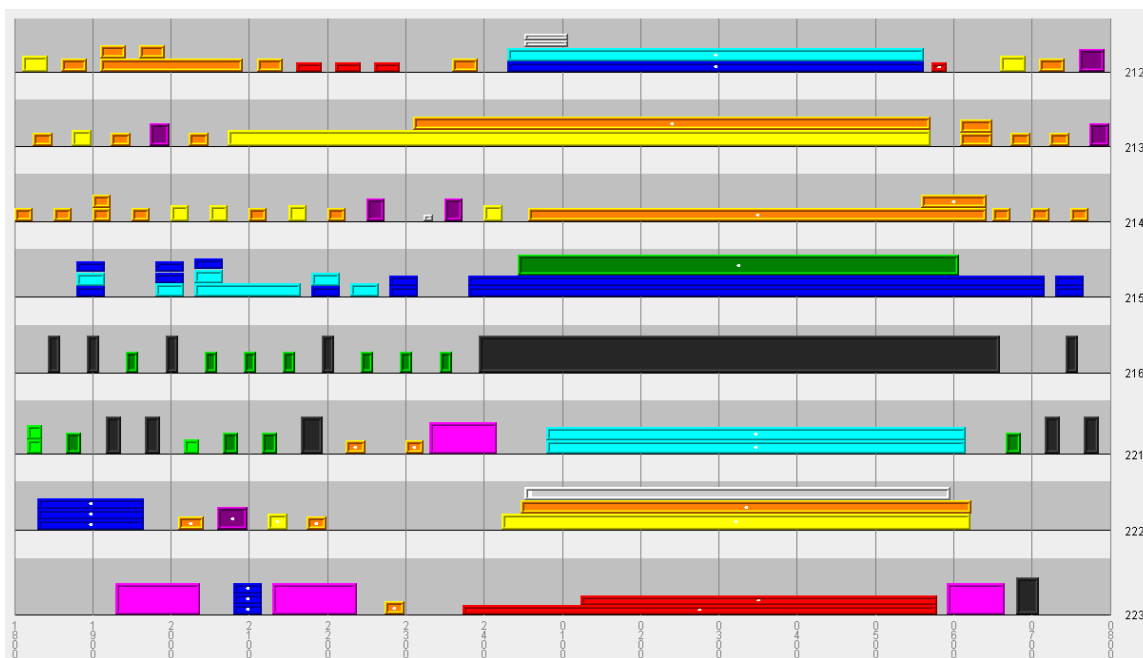


Figure 6.2: Shunt schedule for Hoofddorp from Tuesday 18:00 to Wednesday 08:00

the length of the block represents the planned horizon. The colored blocks represent train units, each color is a different subtype. The lengths of these colored blocks represent the time units are parked on the shunt tracks, and the height show how much track capacity units use. This diagram gives a good overview of the obtained solution. The small white dots on some of the colored blocks show if a unit is not parked on its preferred track.

Table 6.6 shows the results for the third Hoofddorp experiment. In one of our requirements we state that the computation time should be at most 3 hours. When we tried this timetable on the MIP approach, it took almost 3 hours before the model had found a feasible solution. We have continued the search for 11 hours until the stop criterion of 14 hours was reached. Computing even longer would probably result in a better solution, however 14 hours of computation time is already 5 times too long according to our requirements. Our approach did also have some trouble with this computation time. Because we are searching in a much larger search space than when we build a schedule for one day, we had to increase the population size. However the strongest contribution to the increase of the computation time is caused by the simulation model. The model has to simulate almost 5 times more activities in each evaluation since this time 514 units arrive and depart in the timetable compared to the 118 in experiment 1 and 2. To stay in the time limit of 3 hours, we could not increase the population size and the number of generations too much, therefore we did not find a feasible solution in each of the 10 runs, 4 obtained solutions were infeasible, violating between 1 and 3 restrictions. However the feasible solutions that were found all outperformed the MIP approach on each objective and were found in almost  $\frac{1}{5}$  of the MIP's computation time.

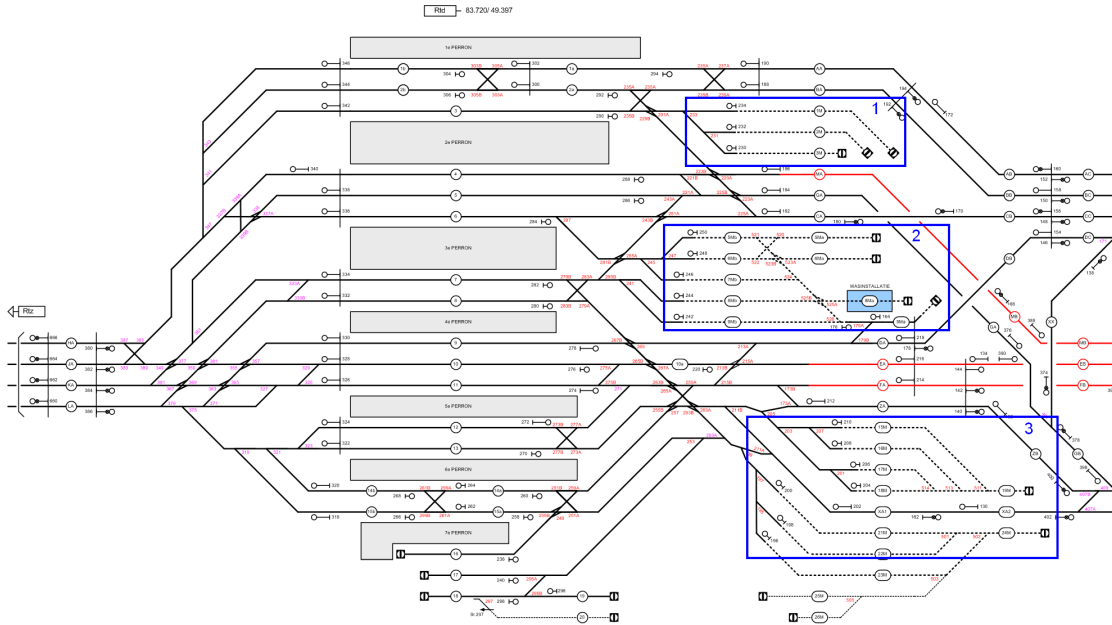
Table 6.6: Result for Hoofddorp experiment 3

	Experiment 3			MIP
	best	worst	average	
Shunt moves	967	970	968.7	972
Arrival breaks	2	4	3.2	3
Departure breaks	2	3	2.5	6
Not on preferred Track	162	133	140.4	223
Not on empty Track	60	65	70.2	104
Capacity penalties	0	2	0.7	0
Crossing penalties	0	1	0.2	0
Infrastructure penalties	0	0	0	0
Int. Train penalties	0	0	0	0
Total fitness	5697	6103	5914.9	6390
Computation time in hours	3	3	3	14

## 6.2 Rotterdam

In the second real world case we are going to apply our method to the shunting problem in Rotterdam. As the second city in the Netherlands, Rotterdam has a huge station with a lot of train activities during the day. The station has 18 different arrival and departure platforms and 14 shunt tracks, see figure 6.3. Trains arrive and depart at the station from both sides, some trains will continue their service in the same direction after a small stop while for other trains Rotterdam is the final stop. Trains from this last group are either shunted to the shunt yard or are used for the reverse train service back to where they came from. At the end of the rush hours those trains can also leave one or more units at the station while the rest of the composition stays in operation. Because of all these different trains leaving in opposite directions and because of the limited space at the shunt yard, Rotterdam is considered to be a difficult location to plan good shunt schedules. Contrary to Hoofddorp where we did not have arrival and departure platforms and all shunt times were fixed, this time flexible shunt times could be very interesting. In the schedules of the local shunt planners we see that they sometimes use this flexibility to reduce the workload during busy hours.

**Rtd preferences** In Rotterdam the shunt yard is divided into 3 separate shunt yards. They are all on the same side of the station but have different routes to the station. Instead of assigning preferred tracks to each train series as we did in Hoofddorp, this time we will have preferred shunt yards on which each track is equally preferred. Depending on the platform a unit arrives or departs, the yard which has the easiest route is preferred over the others. Table 6.7 shows the preferred yard for each platform.



ST	1M	2M	3M	5M	6M	7M	8M	9M	15M	16M	17M	18M	21M	22M
L	320	275	175	155	155	259	325	379	345	320	291	291	259	240

Figure 6.3: Shunt yard at Rotterdam (ST = Shunt track, L = Length in meters)

Table 6.7: Shunt yard preferences in Rotterdam

Platform track		1	3	4	6	7	8	9	10	11	12	13	14	15	16	17
Best shunt yard		1	1	2	2	2	2	3	3	3	3	3	3	3	3	3
Second best shunt yard		2	2	3	3	3	3	2	2	2	2	2	2	2	2	2

### 6.2.1 Experiments

For the Rotterdam case we have performed two different experiments, one with flexible shunt times and one without. This way we can analyse the impact on the performance of the newly introduced flexible shunt times. Table 6.8 shows the settings we have used for these 2 experiments.

In these two experiments we are going to find a shunt schedule for an entire day. From Tuesday 07:00 to Wednesday 08:00. During this time interval, a total of 961 units arrive at the station in 620 different train compositions. Only 71 of those 961 units will have to be shunted, the other units will only make a small stop at the station. 9 different train unit subtypes are parked in Rotterdam, the same as the first 9 units in Hoofddorp which can be found in table 6.1.

Because the Rotterdam case is a bit less constrained than the Hoofddorp case (e.g. more routes to the shunt tracks and no international trains) we have set the penalty weights very close to the weights of the objectives. By choosing very close weights, the chance of getting stuck in local optima is reduced because promising infeasible solutions will also survive in



the population. In the experiment with flexible shunt times, also the waiting time has a weight. We are trying to find better solutions by changing the shunt times, however we do not want units to wait on a platform track for no reason. Therefore, waiting on platform tracks is penalized with a very small weight: if no benefit is obtained by this waiting, it is better not to wait at all. The shunt time experiment has more generations to complete its search because by introducing shunt times the search space has become much larger and more time is needed to converge to a good solution. Since parking on empty tracks is not considered preferable by the planner in Rotterdam, this weight is 0. Table 6.8 shows the settings used in the two experiments.

Table 6.8: Settings for the Rotterdam experiments

Setting without flexible shunt times					
Objective	$w$	Restrictions	$w$	GA parameters	
Shunt move	1	Capacity	13	Generations	500
Arrival break	5	Crossing	13	Population Size	800
Departure break	10	Infrastructure	13	Crossover Rate	0.9
Not on preferred track	1			Mutation Rate	0.9
Not on empty track	0			Elitism	80
Wait time on platform	0				

Setting with flexible shunt times					
Objective	$w$	Restrictions	$w$	GA parameters	
Shunt move	1	Capacity	13	Generations	800
Arrival break	5	Crossing	13	Population Size	800
Departure break	10	Infrastructure	13	Crossover Rate	0.9
Not on preferred track	1			Mutation Rate	0.9
Not on empty track	0			Elitism	80
Wait time on platform	0.001				

## 6.2.2 Results

The results for the Rotterdam experiments can not be compared to results obtained by the MIP approach. The MIP model still needs further development to cope with the complexity of this location so no comparable experiments have been carried out. The changes in some of the compositions during the day is still implemented incorrectly in the MIP model due to the assumption of fixed shunt times. At this point the MIP model will still shunt a single unit of an arriving composition with multiple units, directly after its arrival, even if the path to the shunt yard is blocked by the other units in the composition. In our model we can easily force those units to wait until the train that is blocking the unit has departed. Even if we do not use flexible shunt times, we can change the only possible shunt moment to directly after the departure of the train, instead of directly after its arrival.

**Results without flexible shunt times** Table 6.9 shows the results for the experiments for the Rotterdam case. In both cases the model has no trouble finding feasible solutions.

Table 6.9: Result for Rotterdam experiments 1 & 2

	Without Time			With time		
	best	worst	average	best	worst	average
Shunt moves	125	128	126.6	69	74	72.2
Arrival breaks	1	2	1.2	0	1	0.4
Departure breaks	0	2	1.4	0	1	0.8
Not on preferred Yard	9	21	15.4	7	20	12.4
Wait Time on platform	0	0	0	2024	1920	1988.2
Capacity penalties	0	0	0	0	0	0
Crossing penalties	0	0	0	0	0	0
Infrastructure penalties	0	0	0	0	0	0
Total fitness	139	179	162.1	78.02	110.2	96.59
Computation time in seconds	2021	2018	2020.2	3214	3234	3222.2

The results for the experiments without flexible shunt times appeared to be very good. In the best result only 1 arriving train needs to be broken, and most units are parked on their preferred track. The absolute optimal solution for this case would be if the number of shunt moves was 124 (the least movements needed if no arriving/departing train has to be broken) and all the other objectives would be zero. Based on the results we can conclude that our approach was able to produce solutions very close to this optimum, and we do not even know if this absolute optimum is even in the feasible region of the search space. Figure 6.4 shows the visualisation of the best found shunt schedule.

**Results with flexible shunt times** By introducing flexible shunt times, the results show that we can save a lot of the 124 shunt moves that was the absolute minimum in the case without flexible times. The best solution found, with a total of 69 shunt moves, has reduced the number of shunt moves with 45% compared with the best solution found without shunt times. Also the track preference objective was improved and no arrival or departure break was recorded. From these results it seems that we have created a shunt schedule that takes almost half the work to execute. Before we can really conclude this outcome, we first have to take a better look at the obtained solution, see figure 6.5 for a visualisation of this best found shunt schedule with flexible shunt times.

In section 5.1 we introduced 4 situations in which flexible shunt times could be beneficial for the shunt schedule. The first situation is a train unit that waits on the platform track, until it can be used for a departure at that same track. This way we save up to 2 shunt moves, one for shunting to the shunt yard and 1 for shunting back to the platform. In our solution this happens 24 times, counting for 47 of the 56 saved shunt moves. This counts for the major part of our shunt move savings, however the question is whether flexible shunt times are needed in this case. Those matchings can also be derived from the data, by looking for arrivals and departures after another on the same track with the same unit type. But still it is nice to see that the shunt times were chosen correctly by the GA to create these matches. The more interesting savings are the remaining 9 shunt moves that were

saved. In 6 cases the model found combinations of multiple units retrieved from a shunt track for different departing trains, saving 6 shunt moves. And in 3 cases the model shunted units from different arriving trains together to the shunt track, saving 3 shunt moves.

Still one question remains: do we want to apply all these savings in a real-world schedule. We compared the results with actual planned schedules in Rotterdam and we found out that these typical shunt move savings rarely appeared in the schedules. Only in the first category we found some similarity for units with a small waiting time while in the second category only 1 or 2 of these situations appeared in a complete day schedule. When we discussed our findings with a planner in Rotterdam, it became clear that parking train units for a longer period of time on platform tracks isn't always desirable. The main reason is that it is confusing for the passengers if trains are waiting next to a platform when the trains are not in service. He did point out that it is sometimes necessary when the routes to the shunt yards are heavily occupied, but parking units on platform tracks to save shunt moves was not really desirable. However that did not mean that the whole idea of saving shunt moves in this way was not interesting. Further research on the impact of these decisions on the day to day passengers flow at the station is needed before we can conclude whether these improvements are really desirable or not.

Figure 6.4: Shunt schedule for Rotterdam from Tuesday 07:00

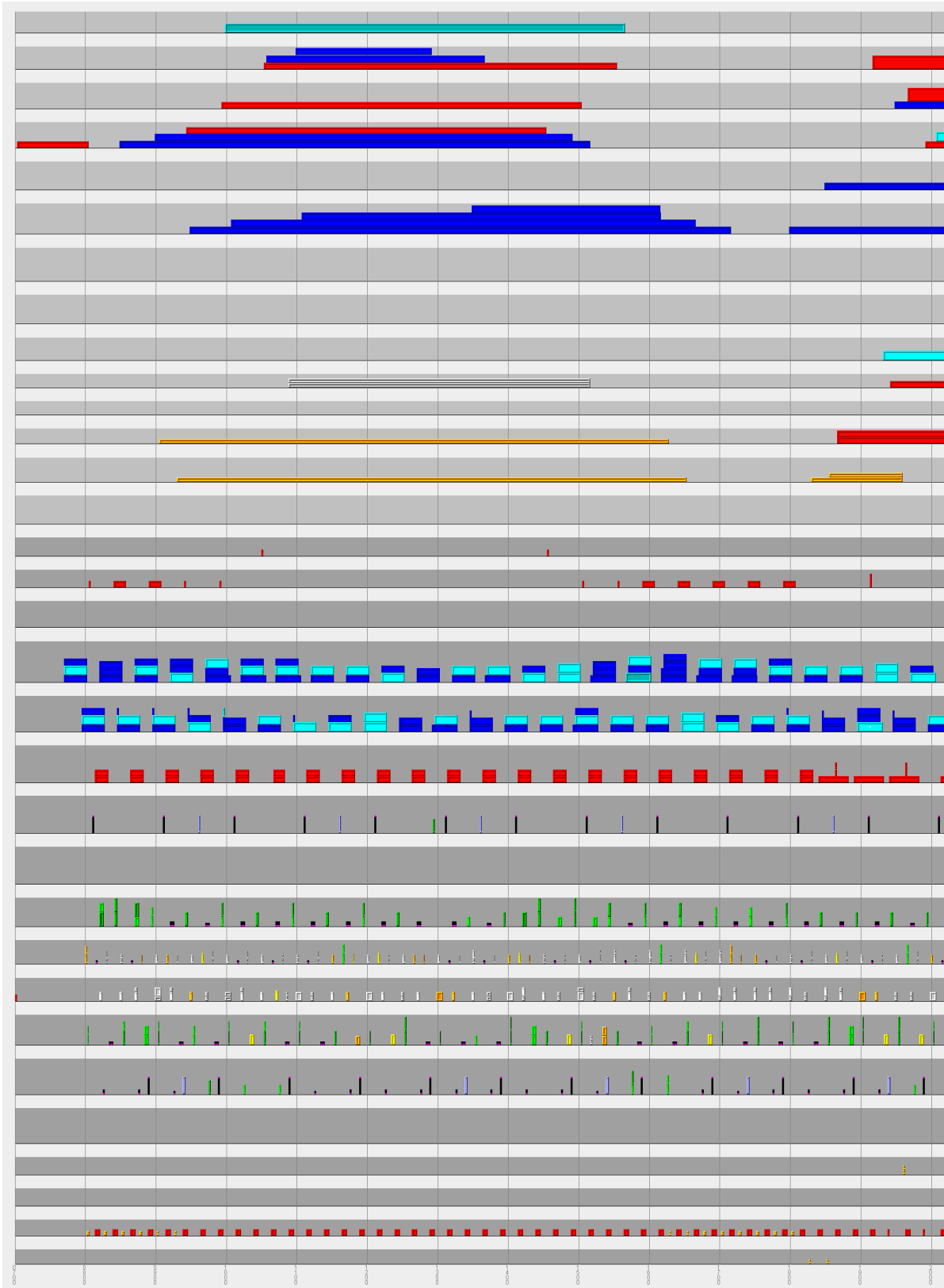


Figure 6.4: to Wednesday 0:800 without time flexibility

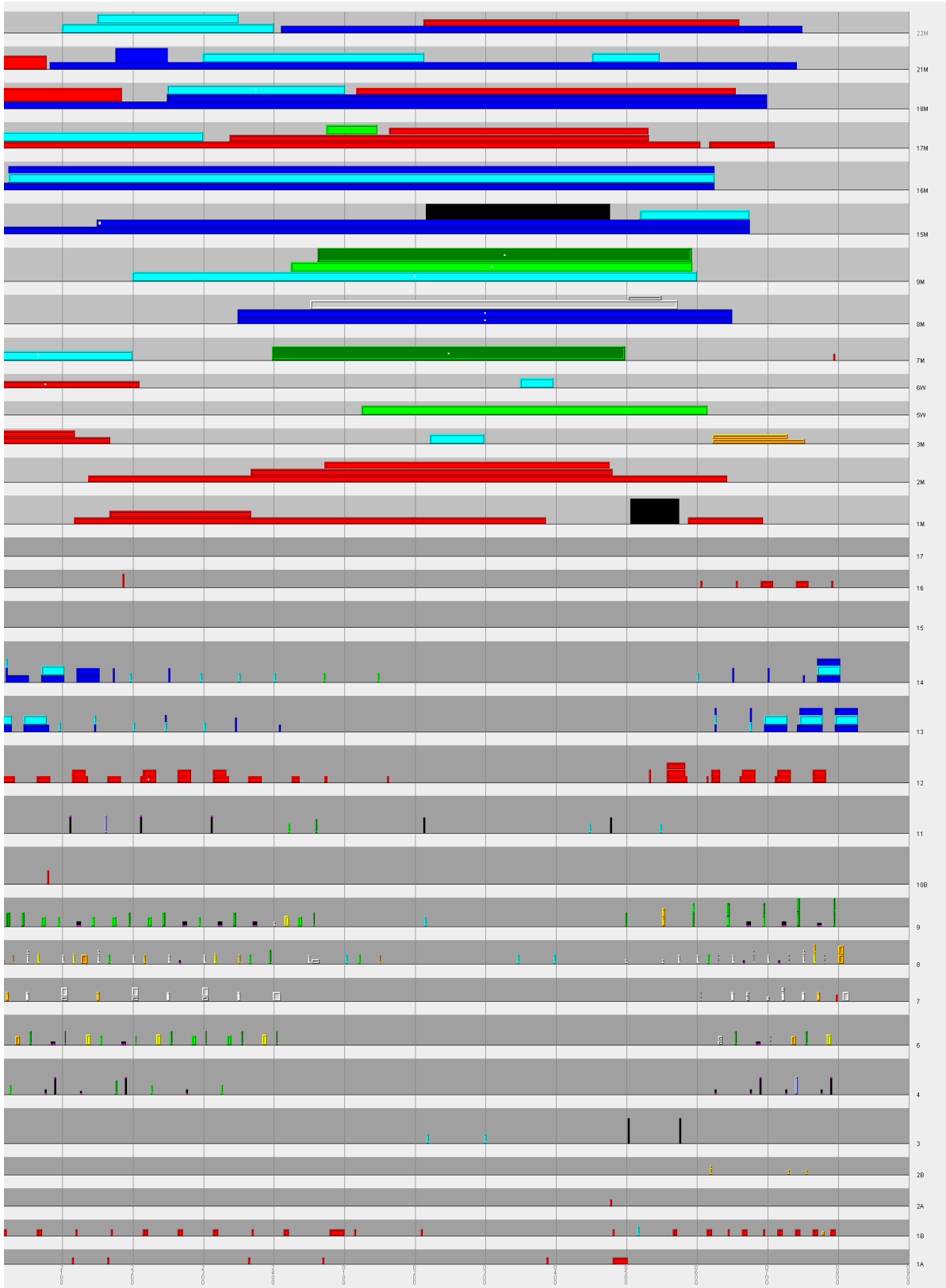


Figure 6.5: Shunt schedule for Rotterdam from Tuesday 07:00

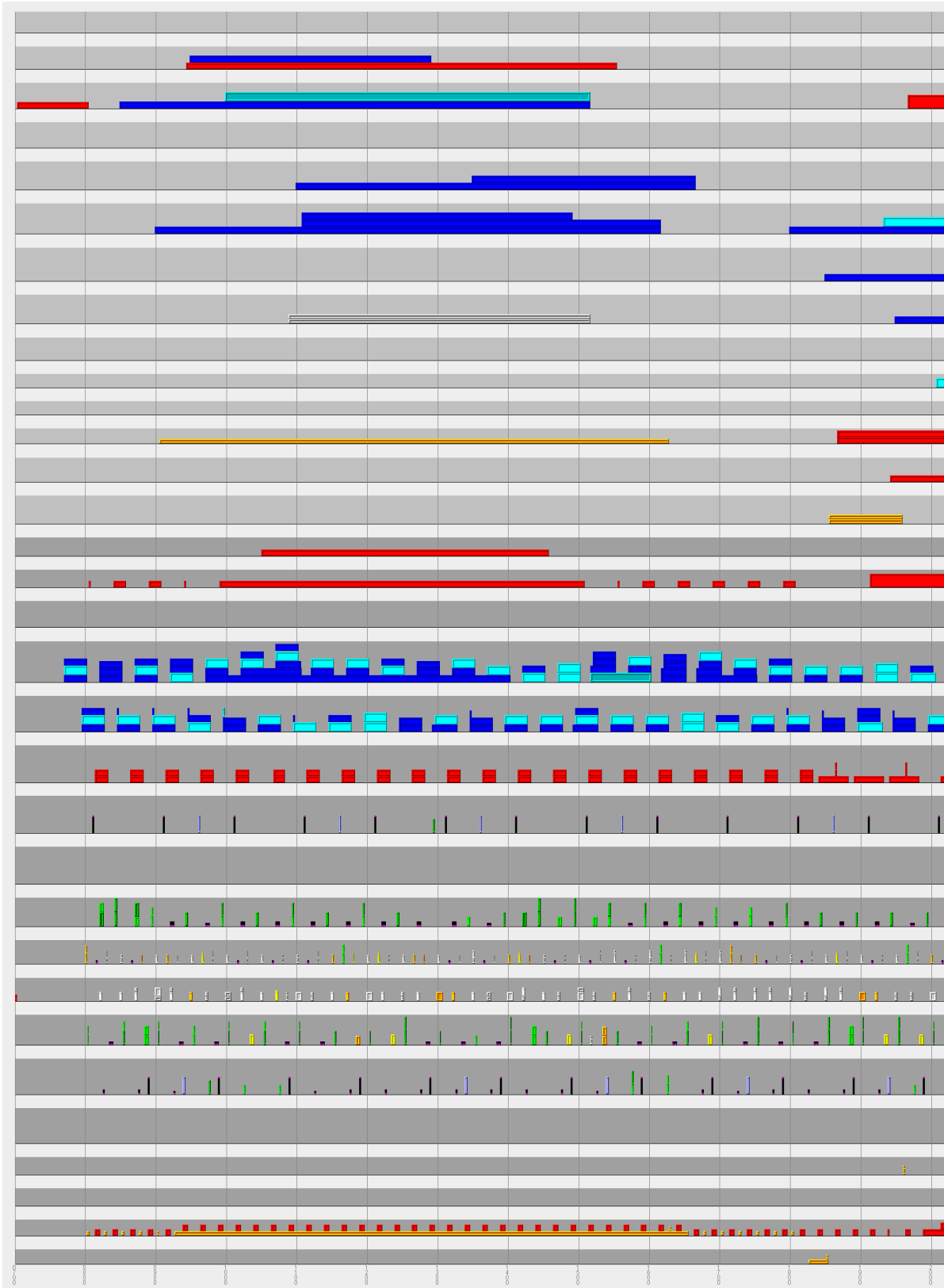
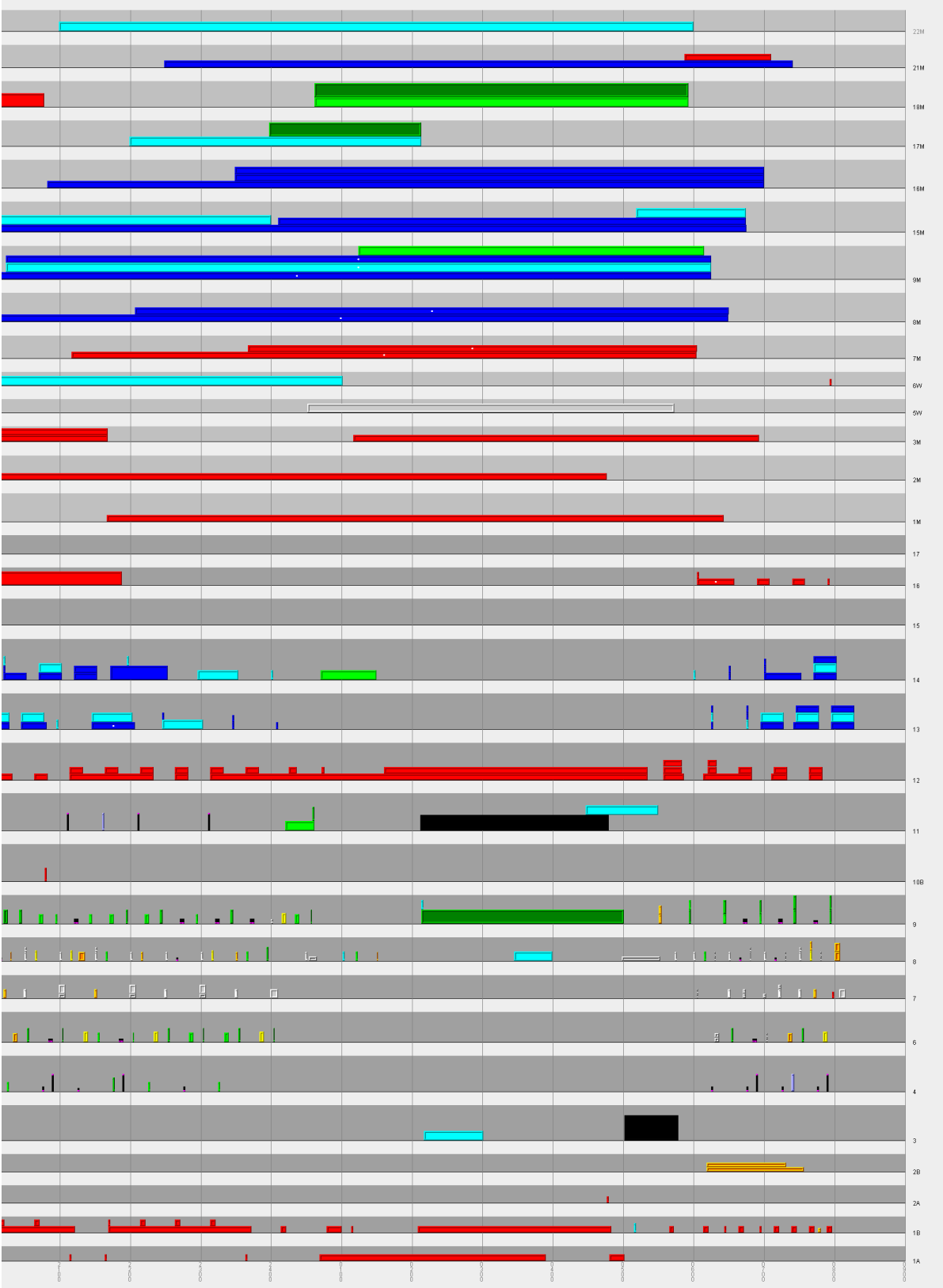


Figure 6.5: to Wednesday 08:00 with time flexibility



### 6.3 Applicability of the solution approach in real-world decision support

In the future Netherlands Railways wants to provide their local planners with a new decision support system that helps these planners with the day-to-day scheduling activities at local planning offices. One part of this local planning consists of the shunt planning at around 30 shunt locations throughout the Netherlands. As part of this decision support system NS is searching for a tool for automated shunt planning. In this scope we have developed a new method for solving the TUSP. The results described in the previous sections are very promising, but the applied method also has some drawbacks. Therefore we want to give a small review of the suitability of our approach in a decision support system.

The major drawback of using metaheuristic approaches is often the various parameters that have to be set by the user for each new problem that is encountered. This also applies when using genetic algorithms. In our research we tried to keep most parameters the same during different test cases, however to obtain the best results, changes in the GA parameters as well as the weights in the fitness functions were needed. Because the shunting problem differs at each of the 30 locations and changes in the timetable occur frequently, this would mean that the planners using this tool would have to adjust the settings themselves. This is something that is really undesirable in today's decision support systems.

An other drawback is the randomness of the genetic algorithm. Each time we run the optimization on exactly the same problem instance, we can get different solutions. The solutions can sometimes be better than any solution found before, but they can also be a lot worse. The planners that will have to work with a new automated tool will need to get confidence in the quality of the provided solutions before they fully adopt the functionality of such a tool into their day to day work. A method that provides different solutions each time they “push the button” will not help this adoption, even if the presented solutions are of good quality.

Applying metaheuristics in the field of Operation Research is relatively new and these techniques are often criticised for their drawbacks. However it is debatable whether the drawbacks really outweighs the benefits. In our case we were able to model the shunting process in much more detail than the current MIP approach. This means that the solutions obtained by the MIP model are based on a lot more assumptions and thus need a thorough evaluation and probably a lot of modifications by the planners. In our approach the planners will have to learn to use the model and understand what kind of impact the weights and settings have on the generated solutions. However, the obtained solutions will be much closer to the schedules used in practice and thus less adjustments should be necessary.



## Chapter 7

# Conclusion

In this research we have studied the problem of train shunt planning. Currently shunt plans are made entirely by hand which is a hard and labour intensive job. To support local planners, Netherlands Railways is developing a new decision support system in which automated shunt scheduling should be one of the features. The aim of this research was to develop a tool that can create feasible and good shunt schedules. To reach this goal we have analyzed current approaches in the literature, and in more detail a MIP approach currently in development at NS. Based on these findings we have developed a new approach that is able to tackle some of the problems that the current approaches are still facing. One of these problems was the assumption of fixed shunt times. In our approach we used a genetic algorithm as search technique and a chromosome representation has been developed that includes flexible shunt times into the search space. We have developed two different approaches: one in which the entire problem was solved integral and one in which the GA is supported by a simple heuristic. Tests on several theoretical cases showed that the combined GA with heuristic approach was superior to the integral approach. With this combined approach we have successfully solved two real-world shunting cases. Compared with the MIP approach, the results were comparable on the smaller problems and much better in the case of larger problem instances. Introducing flexible shunt times turned out to be quite promising. The algorithm had no problems with finding good solutions in the enormously increased search space. In the real-world case of Rotterdam it turned out that a lot of shunt movements can be saved by combining the shunt activities of train units from different trains. These combinations were found by changing the shunt times.

Although we have managed to model the shunting problem in more detail and the results are better than those found by other approaches, it is still debatable if our approach is suitable for real-world decision support. For this purpose you want a “one push on the button” approach without adjusting parameters and other settings. Unfortunately genetic algorithms do need those adjustments as with many other metaheuristics. Still we have proven that a metaheuristic approach is able to solve the highly constrained multi-objective shunt problem better than traditional mathematical models and further research along this path could be very interesting.

**Future research** We have shown that combining the genetic algorithm with a very simple heuristic can lead to great improvements in the results. However our rules are very simple and use only local information. It could be interesting to develop a smarter heuristic or even combine the GA with some LP/MIP method to obtain better results and prevent local optima. An other interesting approach is the so called hybrid genetic algorithm in which some of the individuals in the population are further optimized by some local search technique during the evolution process. To overcome the problem of choosing the right parameters, several techniques have been developed for automatic parameter setting. It could be interesting to apply such techniques on our problem to make it more suitable for real-world decision support.

# Bibliography

- [1] E. Abbink. Intelligent shunting: Dealing with constraints (satisfaction). In W. van Wezel, R. Jorna, and A. Meystel, editors, *Planning in Intelligent Systems: Aspects, Motivations, and Methods*, pages 437–475. John Wiley & Sons, 2006.
- [2] U. Aickelin. *Genetic Algorithms for Multiple-Choice Optimisation Problems*. PhD thesis, European Business Management School - University of Wales Swansea, September 1999.
- [3] U. Aickelin. An indirect genetic algorithm for set covering problems. *Journal of the Operational Research Society*, 53(20):1118–1126, 2002.
- [4] U. Aickelin and K. A. Dowsland. Exploiting problem structure in a genetic algorithm approach to a nurse rostering problem. *Journal of Scheduling*, 3:139–153, 2000.
- [5] U. Aickelin and K. A. Dowsland. An indirect genetic algorithm for a nurse-scheduling problem. *Computers & Operations Research*, 31:761–778, 2004.
- [6] J. Alcaraz, C. Maroto, and R. Ruiz. Solving the multi-mode resource-constrained project scheduling problem with genetic algorithms. *Journal of the Operational Research Society*, 54(6):614–626, 2003.
- [7] S. Bertel and J. C. Billaut. A genetic algorithm for an industrial multiprocessor flow shop scheduling problem with recirculation. *European Journal of Operational Research*, 159:651–662, 2004.
- [8] U. Blasum, M. R. Bussieck, W. Hochstättler, C. Moll, H.-H. Scheel, and T. Winter. Scheduling trams in the morning. *Mathematical Methods of Operations Research*, 49(1):137–148, March 1999.
- [9] X. Cai, D. C. McKinney, and L. S. Lasdon. Solving nonlinear water management models using a combined genetic algorithm and linear programming approach. *Advances in Water Resources*, 24:667–676, 2001.
- [10] S. Cornelsen and G. D. Stefanoo. Platform assignment. In F. Geraets, L. Kroon, A. Schoebel, D. Wagner, and C. Zarolaigiis, editors, *Algorithmic Methods for Railway Optimization*, pages 233–245. Springer, June 2004.
- [11] F. D. Croce, R. Tadei, and G. Volta. A genetic algorithm for the job shop problem. *Computers & Operations Research*, 22(1):15–24, January 1995.

- [12] C. R. Darwin. *On the origin of species by means of natural selection, or the preservation of favoured races in the struggle of life*. John Murray, London, 1 edition, 1859. Retrieved from <http://darwin-online.org.uk>.
- [13] K. Deb. An efficient constraint handling method for genetic algorithms. *Computer Methods in Applied Mechanics and Engineering*, 186(2-4):311–338, June 2000.
- [14] R. Freling, R. M. Lentink, L. G. Kroon, and D. Huisman. Shunting of passenger train units in a railway station. *Transportation Science*, 39(2):261–272, May 2005.
- [15] G. Gallo and F. D. Miele. Dispatching buses in parking depots. *Transportation Science*, 35(3):322–330, June 2001.
- [16] J. F. Gonçalves, J. J. de Magalhães Mendes, and M. G. C. Resende. A hybrid genetic algorithm for the job shop scheduling problem. *European Journal of Operational Research*, 167(1):77–95, November 2005.
- [17] R. Haijema, C. Duin, and N. van Dijk. Train shunting: A practical heuristic inspired by dynamic programming. In W. van Wezel, R. Jorna, and A. Meystel, editors, *Planning in Intelligent Systems: Aspects, Motivations, and Methods*, pages 437–475. John Wiley & Sons, 2006.
- [18] S. He, R. Song, and H. Anzhou. Application of genetic algorithm in computer-aided dispatching problem for railyards operations. In *Proceedings of International Conference on Artificial Intelligence for Engineering*, 1998.
- [19] S. He, R. Song, and S. S. Chaudry. Fuzzy dispatching model and genetic algorithms for railyards operations. *European Journal of Operational Research*, 124(3):307–331, August 2000.
- [20] ILOG, <http://www.ilog.com/products/cplex/>. *ILOG CPLEX*.
- [21] L. G. Kroon, R. M. Lentink, and A. Schrijver. Shunting of passenger train units: an intergrated approach. Technical report, Erasmus Univeristy Rotterdam, 2007.
- [22] R. M. Lentink, P.-J. Fioole, L. G. Kroon, and C. van’t Woudt. Applying operations research techniques to planning of train shunting. In W. van Wezel, R. Jorna, and A. Meystel, editors, *Planning in Intelligent Systems: Aspects, Motivations, and Methods*, pages 415–436. John Wiley & Sons, 2006.
- [23] T.-K. Liu, J.-T. Tsai, and J.-H. Chou. Improved genetic algorithm for the job-shop scheduling problem. *The International Journal of Advanced Manufacturing Technology*, 27(9-10):1021–1029, February 2006.
- [24] Z. Michalewicz and M. Schoenauer. Evolutionary algorithms for constrained parameter optimization problems. *Evolutionary Computation*, 4(1):1–32, 1999.
- [25] M. Mori and C. C. Tseng. A genetic algorithm for multi-mode resource constrained project scheduling problem. *European Journal of Operational Research*, 100(1):134–141, July 1997.

- [26] Netherlands Railways. *Annual Report 2008*, 2009.
- [27] C. C. Palmer and A. Kershenbaum. Representing trees in genetic algorithms. In *Proceedings of the First IEEE International Conference on Evolutionary Computation*, pages 379–384, June 1994.
- [28] C. C. Ribeiro and P. Hansen. *Essays and Surveys in Metaheuristics*, volume 1 of *Operation Research - Computer Science Interface Series*. Kluwer Academic, January 2001.
- [29] R. G. Sargent. Verification and validation of simulation models. In S. Mason, R. Hill, L. Mönch, O. Rose, J. Jefferson, and J. Fowler, editors, *Proceedings of the 2008 Winter Simulation Conference*, pages 157–169, 2008.
- [30] G. D. Stefano and M. L. Koči. A graph theoretical approach to the shunting problem. *Electric Notes in Theoretical Computer Science*, 92:16–33, 2004.
- [31] N. Tomii and L. J. Zhou. Depot shunting scheduling using combined genetic algorithm and pert. In J. Allan, R. J. Hill, C. A. Brerbia, G. Sciutto, and S. Sone, editors, *Computers in Railways VII: Proceedings COMPRAIL 2000*, pages 437–446. WIT Press, 2000.
- [32] N. Tomii, L. J. Zhou, and N. Fukumara. An algorithm for station shunting scheduling problems combining probabilistic local search and pert. In I. Imam, Y. Kodratoff, A. El-Dessouki, and M. Ali, editors, *Multiple Approaches to Intelligent Systems*, pages 788–797. Springer, June 1999. 12th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems.
- [33] W. van Wezel. Task oriented support for train shunting planning. In J. R. Wilson, T. Clarke, and A. Mills, editors, *People and Rail Systems: Human Factors at the Heart of the Railway*, pages 309–318. Ashgate, August 2007.
- [34] T. Winter and U. T. Zimmermann. Real-time dispatch of trams in storage yards. *Annals of Operations Research*, 96(1-4):287–315, November 2000.
- [35] X.-S. Yang. *Nature-Inspired Metaheuristic Algorithms*. Luniver Press, 2008.

## Appendix A

# Crossover Experiments

In this experiment we have tested 3 different crossover operators; the one-point, two-point and the uniform crossover. In the one-point crossover, one random point in the chromosome is chosen. At this point the chromosomes of the 2 parents is split into two parts. The offspring's chromosome copies the left part of the chromosome of the first parent and the right part of the chromosome of the second parent. 2 random points are chosen in the two-point crossover. The parents chromosomes are split into three parts at these points. The offspring's chromosome copies the first and third part of the first parent and the second part of the second parent. In the uniform crossover, each allele in the offspring's chromosome is randomly copied from either the first or the second parent. In this experiment we use the third theoretical case described in section 5.1, to test each of the three crossover operators. Table A.1 shows the used settings of the experiments.

Table A.1: Settings for theoretical case 2, crossover experiments

Setting for crossover experiments			
Objectives	Weights	Settings	
Shunt moves	1	Number of generations	200
Arrival breaks	2	Population size	100
Departure breaks	5	Crossover rate	0.9
Capacity penalty	10	Mutation rate	0.9
Crossing penalty	10	Elitism	10

Each crossover is tested 10 times, table A.2 shows the results. The one and two-point crossover perform quite similar, with the one-point crossover slightly better. The uniform crossover performance clearly worse than the first two crossover operators.

Table A.2: Result different crossover operators

	over 10 runs	Shunt moves	Arrival breaks	Departure breaks	Capacity penalties	Crossing penalties	Fitness
one point	best solution	153	38	18	0	0	319
	worst solution	165	45	23	0	0	370
	average	158.7	41.7	20	0	0	343.1
two point	best solution	156	40	19	0	0	331
	worst solution	165	44	24	0	0	373
	average	160.7	42.4	21.3	0	0	352
uniform	best solution	162	43	22	0	0	358
	worst solution	172	47	28	0	1	416
	average	166.1	43.8	25.2	0	0.3	382.6

# Appendix B

## Timetables

Timetables for theoretical case 2, 3 and shunt location Hoofddorp



Table B.1: Timetable for theoretical case 2

Train Nr	Arrive/Depart	A/D side	Time	Composition	Platform
1	A	B	16:40	A	p1
2	A	B	16:45	A	p1
3	A	B	16:50	A	p1
4	A	B	16:55	A	p1
5	A	B	17:00	A	p1
6	A	B	17:05	B	p1
7	A	B	17:10	B	p1
8	A	B	17:15	B	p1
9	A	B	17:20	B	p1
10	A	B	17:25	B	p1
11	A	B	17:30	C	p1
12	A	B	17:35	C	p1
13	A	B	17:40	C	p1
14	A	B	17:45	C	p1
15	A	B	17:50	C	p1
16	A	B	17:55	D	p1
17	A	B	18:00	D	p1
18	A	B	18:05	D	p1
19	A	B	18:10	D	p1
20	A	B	18:15	D	p1
21	A	B	18:20	E	p1
22	A	B	18:25	E	p1
23	A	B	18:30	E	p1
24	A	B	18:35	E	p1
25	A	B	18:40	E	p1
100	V	B	19:10	EDCBA	p1
200	V	B	20:40	EDCBA	p1
300	V	B	21:10	EDCBA	p1
400	V	B	22:40	EDCBA	p1
500	V	B	23:10	EDCBA	p1

Table B.2: Timetable for theoretical case 3, part 1: hundred units arrive

Train Nr	Arrive/Depart	A/D side	Time	Composition	Platform
1001	A	B	10:00	CCA	p1
1002	A	B	10:05	BA	p1
1003	A	B	10:10	ABE	p1
1004	A	B	10:15	BBD	p1
1005	A	B	10:20	EFE	p1
1006	A	B	10:25	DEA	p1
1007	A	B	10:30	DE	p1
1008	A	B	10:35	BB	p1
1009	A	B	10:40	A	p1
1010	A	B	10:45	BFE	p1
1011	A	B	10:50	BF	p1
1012	A	B	10:55	FCF	p1
1013	A	B	11:00	CF	p1
1014	A	B	11:05	A	p1
1015	A	B	11:10	A	p1
1016	A	B	11:15	C	p1
1017	A	B	11:20	D	p1
1018	A	B	11:25	DAD	p1
1019	A	B	11:30	B	p1
1020	A	B	11:35	BFC	p1
1021	A	B	11:40	E	p1
1022	A	B	11:45	C	p1
1023	A	B	11:50	EC	p1
1024	A	B	11:55	FEC	p1
1025	A	B	12:00	DD	p1
1026	A	B	12:05	D	p1
1027	A	B	12:10	BD	p1
1028	A	B	12:15	E	p1
1029	A	B	12:20	DEA	p1
1030	A	B	12:25	ACB	p1
1031	A	B	12:30	BC	p1
1032	A	B	12:35	BC	p1
1033	A	B	12:40	BA	p1
1034	A	B	12:45	CF	p1
1035	A	B	12:50	F	p1
1036	A	B	12:55	CED	p1
1037	A	B	13:00	E	p1
1038	A	B	13:05	CFF	p1
1039	A	B	13:10	C	p1
1040	A	B	13:15	BFF	p1
1041	A	B	13:20	EDE	p1
1042	A	B	13:25	CC	p1
1043	A	B	13:30	CCB	p1
1044	A	B	13:35	AE	p1
1045	A	B	13:40	FEF	p1
1046	A	B	13:45	FAF	p1
1047	A	B	13:50	D	p1
1048	A	B	13:55	E	p1

Table B.3: Timetable for theoretical case 3, part 2: hundred units depart

Train Nr	Arrive/Depart	A/D side	Time	Composition	Platform
2001	V	B	14:00	FCC	p1
2002	V	B	14:05	E	p1
2003	V	B	14:10	CB	p1
2004	V	B	14:15	BEB	p1
2005	V	B	14:20	D	p1
2006	V	B	14:25	F	p1
2007	V	B	14:30	FDE	p1
2008	V	B	14:35	E	p1
2009	V	B	14:40	C	p1
2010	V	B	14:45	CE	p1
2011	V	B	14:50	FCD	p1
2012	V	B	14:55	EEA	p1
2013	V	B	15:00	F	p1
2014	V	B	15:05	DDF	p1
2015	V	B	15:10	EDE	p1
2016	V	B	15:15	DB	p1
2017	V	B	15:20	BAD	p1
2018	V	B	15:25	D	p1
2019	V	B	15:30	CCD	p1
2020	V	B	15:35	EC	p1
2021	V	B	15:40	DEF	p1
2022	V	B	15:45	AA	p1
2023	V	B	15:50	D	p1
2024	V	B	15:55	AF	p1
2025	V	B	16:00	F	p1
2026	V	B	16:05	AB	p1
2027	V	B	16:10	CC	p1
2028	V	B	16:15	AAA	p1
2029	V	B	16:20	CB	p1
2030	V	B	16:25	BED	p1
2031	V	B	16:30	B	p1
2032	V	B	16:35	FAB	p1
2033	V	B	16:40	D	p1
2034	V	B	16:45	FD	p1
2035	V	B	16:50	F	p1
2036	V	B	16:55	F	p1
2037	V	B	17:00	FAB	p1
2038	V	B	17:05	BCA	p1
2039	V	B	17:10	CEE	p1
2040	V	B	17:15	DC	p1
2041	V	B	17:20	E	p1
2042	V	B	17:25	B	p1
2043	V	B	17:30	CF	p1
2044	V	B	17:35	FE	p1
2045	V	B	17:40	CF	p1
2046	V	B	17:45	CBB	p1
2047	V	B	17:50	EE	p1
2048	V	B	17:55	FBE	p1
2049	V	B	18:00	C	p1

Table B.4: Timetable for Hoofddorp, from Tuesday 18:00 to 20:00

Train Nr	Arrive/Depart	A/D side	Time	Composition	Platform
73365	A	B	1800	MDDM-4	p1
73956	A	B	18:06	DD-AR-4	p1
73158	A	B	18:10	VIRM-4, IRM-4	p1
74358	A	B	18:14	MDDM-4	p1
74367	V	B	18:14	MDDM-4	p1
71658	A	B	18:18	ICM-3, ICM-3, ICM3	p1
73169	V	B	18:22	VIRM-4, IRM-4	p1
73560	A	B	18:26	K-NDC	p1
73696	V	B	18:26	DD-AR-4	p1
73367	A	B	18:30	MDDM-4	p1
73364	V	B	18:30	MDDM-4	p1
73958	A	B	18:36	MDDM-4	p1
73569	V	B	18:36	K-NDC	p1
73160	A	B	18:40	VIRM-6	p1
74360	A	B	18:44	DD-AR-4	p1
74369	V	B	18:44	MDDM-4	p1
70760	A	B	18:48	ICM-3, ICM-4, ICM3	p1
73171	V	B	18:52	VIRM-6	p1
73562	A	B	18:56	K-NDC	p1
73971	V	B	18:56	MDDM-4	p1
73369	A	B	19:00	MDDM-4, MDDM-4	p1
73366	V	B	19:00	DD-AR-4	p1
73960	A	B	19:06	MDDM-4, MDDM4	p1
73571	V	B	19:06	K-NDC	p1
73162	A	B	19:10	K-NDC	p1
70771	V	B	19:10	ICM-3, ICM-4, ICM-3	p1
74362	A	B	19:14	MDDM-4	p1
74371	V	B	19:14	MDDM-4, MDDM-4	p1
73173	V	B	19:22	K-NDC	p1
73564	A	B	19:26	VIRM-6	p1
73973	V	B	19:26	MDDM-4	p1
73368	V	B	19:30	MDDM-4	p1
73371	A	B	19:30	MDDM-4	p1
73573	V	B	19:36	VIRM-4	p1
73962	A	B	19:36	MDDM-4	p1
73164	A	B	19:40	K-NDC	p1
71673	V	B	19:40	ICM-3, ICM-3, ICM-3	p1
74371	V	B	19:44	MDDM-4	p1
70764	A	B	19:48	ICM-4, ICM-3, ICM-3	p1
73175	V	B	19:52	K-NDC	p1
73566	A	B	19:56	K-NDC	p1
73975	V	B	19:52	MDDM-4	p1
73373	A	B	20:00	DD-AR-4	p1