The Train Unit Shunting Problem with Reallocation

Master Thesis Econometrics & Management Science
Specialization Operations Research & Quantitative Logistics

F.E. Wolfhagen
Student number: 362063

First supervisor: Dr. T. Dollevoet
Second supervisor: Prof. dr. D. Huisman
Supervisor NedTrain: Ir. B. Huisman

Erasmus School of Economics
Erasmus University Rotterdam
The Netherlands
March 23, 2017

Abstract

This thesis is concerned with formulating the Train Unit Shunting Problem (TUSP) with Reallocation (TUSP-R) as an MIP and solving it using exact solution methods. The TUSP has been formulated as an MIP in several different ways in previous works, but reallocation was never incorporated. Exactly solving the TUSP-R proved to be very challenging, so a method using a combination of the exact solution method row generation and the heuristic method tabu search was developed which solved the real life instances for shunting yard *Utrecht OZ* in acceptable solution times. The instances for a simplified version of *Kleine Binckhorst* were solved up to a certain problem size; hereafter the computation times became too long for practical use. Our method performs best for problem instances with diverse train types and relatively many different services on a LIFO style shunting yard layout. This keeps the size of the constraint matrix of the initial problem without crossing constraints relatively small compared to other types of problem instances.

Preface

This thesis was written for partial fulfillment of the graduation requirements of the master programme Econometrics and Management Science, with a specialization in Operations Research and Quantitative Logistics at Erasmus School of Economics. I was engaged in the research and writing from October 2016 till March 2017.

This research was undertaken on request of *NedTrain*, where I did an internship during the course of the project. As shunting yard capacity is becoming a more and more urgent topic, the need for an exact formulation for the TUSP-R arose. The research is part of the challenging task to integrate all aspects of the shunting problem, thus also service activities, in one MIP. It also serves as reference, justification, and mathematical background for the current methods used by *NedTrain* to determine shunting yard capacity.

There are a few people without whom I would never have finished this thesis. First of all, I am very grateful to the great guidance I received from dr. Dollevoet from the Econometric department at Erasmus School of Economics. As my supervisor, he provided me with very useful feedback and helped me across bumps along the way. I always left our meetings feeling more motivated and confident than before. His relaxed attitude and precise evaluation of my work greatly contributed to its quality. I also want to thank professor Huisman, who is my second supervisor. His experience at *NS* is a valuable addition to the critical evaluation of my work. Besides the people in Rotterdam, I would like to thank *NedTrain* and especially Bob Huisman -a different mister Huisman- for providing me with the opportunity to write my thesis at the Maintenance Development department. I thoroughly enjoyed working there and many of my kind colleagues were readily available to help me out whenever I needed it. Bob, our discussions on the topic provided me with many valuable insights and alternative viewpoints which helped me try new approaches when I was stuck and brought me further in the process. Besides Bob, Roel van den Broek and Demian de Ruijter also deserve a special thank you for their help. Furthermore, I am very grateful and owe a great deal to my parents for their support during the writing of this thesis and my entire studies. Without them, all of this would never have been possible. Lastly, I would like to thank my friend Elise den Dekker for her moral support during the last phase of the process.

I hope you enjoy reading it, and I advise you to take your time for the mathematical formulations in this thesis. They can be rather complex and potentially frustrating, I know all about it, but definitely have a certain charm to them.

- Floor Wolfhagen, March 2017

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The Netherlands has a rather complex rail network, on which multiple actors involved in both passenger and freight transportation operate. *De Nederlandse Spoorwegen*, translated as the Netherlands Railways and usually referred to by its abbreviation *NS,* is the biggest actor on the passenger transportation market: the company transports over a million passengers a day. It executes about 4800 daily train rides over a network of approximately 2100 km in length. The company owns almost 3000 carriages with 260'000 passenger seats. Besides the *NS*, there are six other passenger transportation operators active on the Dutch railway network. They cover about 700 km of the total network. Most of these passenger train rides take place during the day. Figure 1.1 gives an impression of the complexity and the ramified nature of the network.

Because of the immense numbers mentioned above, most of the rolling stock is in use during the day, especially during the peak hours. At night, when freight transportation takes over, and outside day peak hours, however, passenger rail operators do not use all of their rolling stock. This means that during these times the passenger railway operators have to deal with a surplus of rolling stock. Due to the complex nature of the Dutch network, the surplus cannot be efficiently stalled on tracks that are part of the main network without blocking operations that need to be carried out by other passenger or freight trains. Therefore, the superfluous rolling stock of a rail operator is usually stored in a so-called shunting yard, also for the performance of maintenance activities. Seeing the *NS* owns almost 3000 carriages, this can become quite a puzzle. *NedTrain* is *NS*'s subsidiary in charge of performing these maintenance activities. *NedTrain* is in charge of the trains from the moment they enter the shunting yards and the company is required to perform all necessary maintenance activities before the trains are needed for transportation services again. Besides scheduling all required activities, such as cleaning from the inside, washing from the outside, inspecting, and repairing, fitting all trains on the available tracks and routing them feasibly is a real challenge as well.

Figure 1.1: Railway map of the Netherlands, 2016[1]

Shunting yards have some specific characteristics that make routing and parking so complex. In figure 1.2 a map of shunting yard *Kleine Binckhorst* is displayed. This map shows some of the characteristics a shunting yard may have. First of all, it can be seen that routing over the available tracks is not as straightforward as one might think. Obviously, trains can only move horizontally

[1] www.spoorkaart2016.nl

and can only move over tracks, as opposed to for example cars or trucks. This complicates the matter greatly as it imposes multiple restrictions on the possible movements of the trains. The horizontal orientation implies that some turns cannot be made, even if the tracks are connected to one another. For example, a train cannot go directly from track 52 to track 53 in figure 1.2. It needs to use track 104a. It generally cannot use the diagonal track on the right side of the image to reach track 53, because in order for it to fully exit track 52, it would have to use track 906a (in case of normal length trains). However, this track is part of the main rail network and it is not allowed to use such tracks when not performing passenger duties. This would induce violations of safety regulations. The obvious feature that trains can only move over tracks is especially relevant when multiple trains are parked on the same track. If train A is parked on the left side of a track, and train B on the right side of the track, train A cannot reach the right side of the track without train B having moved out of the way first. In other words, overtaking is not possible. This is perhaps the greatest restrictive property of trains and shunting yards. In figure 1.2, the cleaning and inspecting platforms and tracks are marked, as well as the external washing machine. It can also be seen that some tracks can be approached from both ends, such as tracks 52, 53, etcetera, while others, such as track 104a or 63, can only be approached from one end. The former tracks are referred to as free tracks; the latter as one-sided or LIFO tracks. As can be deduced from the name, the one-sided or LIFO tracks act as Last In First Out (LIFO) stacks.

In this research, we focus solely on the parking and routing of trains in a shunting yard rather than also on the execution of service activities. When considering the parking and routing of trains, we also need to consider coupling and decoupling of train units. Trains can be made up of multiple train units. Train units have different types and only train units with the same type can form a train together. Train types include the *VIRM* and the *SLT*, pictures of which can be found in figure 1.3 and 1.4. The displayed *VIRM* has four carriages, while the *SLT* has six. *NS Reizigers*, the department of *NS* that is in charge of the train schedules, decides the composition of the arriving and departing trains. Sometimes, a certain arriving train unit is assigned to a specific departing service. Other times, it does not matter which train unit is assigned to a departing service, as long as it is of a predefined type with the predefined number of carriages and all required service activities have been performed on it. This means it can be necessary to couple and decouple train units to and from one another during their stay in the shunting yard.

We aim to formulate the described problem as a Mixed Integer Programming Problem (MIP). MIPs for similar problems have been formulated by others, but they all lack one essential property that we do incorporate: the possibility of reallocation. Reallocation can best be explained by means of an example. We consider the train data as depicted in table 1.1.

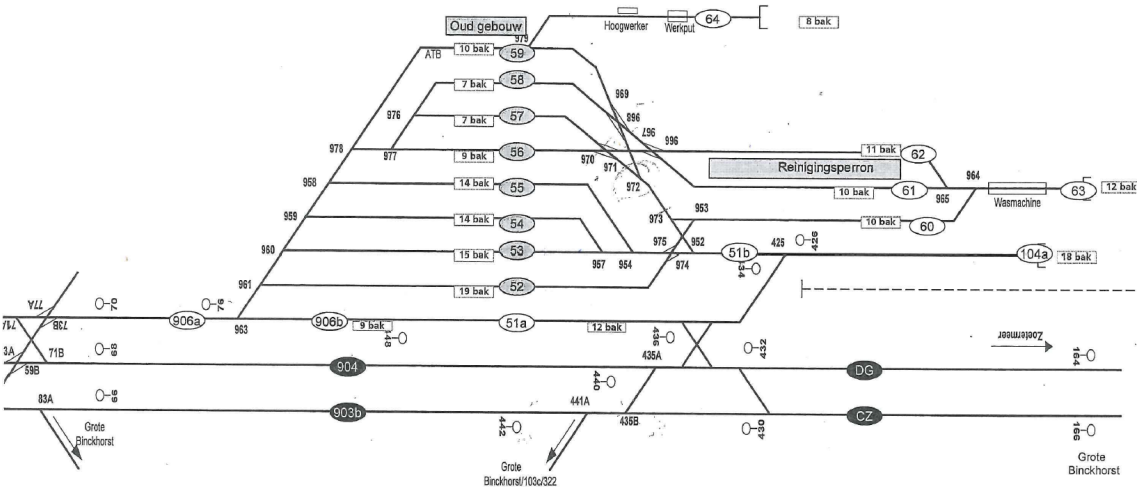


Figure 1.2: Map of Kleine Binckhorst

Figure 1.3: VIRM-4

Figure 1.4: SLT-6

As can be seen in table 1.1, all of the considered trains consist of one train unit only. Furthermore, all of the arriving (departing) train units are of different types (and number of carriages). Therefore, no coupling and decoupling will take place, and the trains all have a fixed arrival and departure time. That is, train 1, arriving at 8:00 consisting of a single train unit with type A, will leave as train 5, consisting of a single train unit with type A at 10:15; train 2, arriving at 8:30 consisting of a single train unit with type B, will leave as train 6, consisting of a single train unit with type B. Similarly, train 3 will arrive at 9:15 and leave as train 7 at 12:00 and train 4 will arrive at 9:40 and leave as train 8 at 12:10.

Now, we take a look at figure 1.5. In this figure, an example shunting yard is laid out and the arriving trains are parked on the tracks. The thick line represents the main rail network, while the thin lines represent the tracks of the actual shunting yard. As can be seen from the figure, this example shunting yard consists of LIFO tracks only. The figure is to scale. Note that in the figure, train 1 and 4 (later on to leave as train 5 and 8, respectively), cannot change tracks without driving back the way they came from, moving up the diagonal connecting track until they fully exit their current track, and effectively using the track that belongs to the main railway network. It does not matter on which track they are parked: they will always need to use this main railway network track to switch tracks. Remember this is not allowed during their stay in the shunting yard. When we look at train 2 (later train 6), we see that it has the possibility to switch between all three tracks without using the main railway network, and train 3 (later train 7) can feasibly move between the bottom two tracks. We can also see in the image that train 2 and 3 need to be parked on the same track, since there is no room to park another train next to either train 1 or train 4. If we park train 2 and 3 on separate tracks, we need to move one of them to the track of the other in order to provide enough space to park train 4. Now, we will get a situation similar to the one in figure 1.5. Comparable situations arise when trains 2 and 3 are parked on the bottom track and train 1 is parked on the middle track. We will now see why we cannot initially park trains 2 and 3 on the top track.

When the trains are leaving, train 5 (formerly train 1) leaves first. Train 6 (train 2) is next in line to move, but train 3 is in the way. Since we concluded earlier on that there was no other way to park the trains than to assign trains 2 and 3 to the same track, there would be no feasible solution if we would not allow train 3 to be moved to another track. If train 2 and 3 were initially parked on the top track, we would not be able to move train 3 to another track, thus rendering no feasible solution. Now, we move train 3 to the former track of train 1, train 6 (formerly train 2) can leave, and consequently train 7 (train 3) and train 8 (train 4) can leave the shunting yard without further ado. If we initially park trains 2 and 3 on different tracks and then move train 2 to the track of train 3 before the arrival of train 4, all trains can leave the shunting yard without

Train unit ID	Train units (displayed by type)	Arrival time	Train unit ID	Train units (displayed by type)	Departure time
1	A	8:00	5	A	10:15
2	B	8:30	6	B	11:05
3	C	9:15	7	C	12:00
4	D	9:40	8	D	12:10

Table 1.1: Example: arriving and departing times and types of trains 1-8

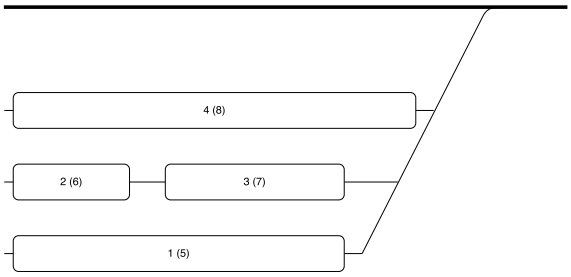


Figure 1.5: Example: need for reallocation

moving the trains in between anymore. Note an exception applies to the upper track: we should not park train 3 here in this example, since train 4 must be parked on this track as it is the only track it fits on. We call the process of moving trains to park them -possibly temporarily- on other tracks than their initial parking tracks reallocation.

As follows from the previous example, reallocation of train units in order to make room for others can be very useful and we will incorporate it in our MIP. Our MIP will be inspired by two more broadly studied problems, for which several MIPs have already been formulated: the Train Unit Shunting Problem (TUSP) and the Rush Hour Problem (RHP). The former is identical to our problem without considering reallocation. This means that the train units need to be initially parked in such a way that they can all leave the shunting yard in the correct configuration without need for reallocation. Reallocation greatly complicates the formulation of our problem with respect to the formulation of the TUSP, since we face the possibility of an almost unlimited amount of variables and/or constraints if we allow an unlimited number of reallocations. The latter refers to the game Rush Hour, which is a children's game in which several 'cars' are located on a grid, and one specific car needs to exit the grid. All cars can only move either vertically or horizontally, and there is one exit, located in line with the car required to exit the grid. The user wants to move the cars in such a way that there is free passing for the specified car to leave the grid. In figure 1.6 an example of the game including possible solution steps is given.

The upper left corner image shows the start configuration of the problem, and the following images show the moves that lead to the solution. In our problem, we deal with a situation in which parked trains are moved around in order to allow for one train to leave the yard without any other trains blocking the way. This contains elements of Rush Hour and the related RHP. In a shunting yard, all trains can move in only one orientation, namely horizontally (we can choose the orientation of

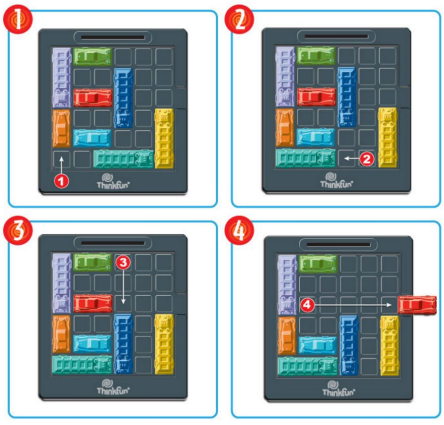


Figure 1.6: Rush Hour instance with solution steps[2]

[2]www.thinkfun.com

our shunting yard ourselves).

Our problem thus contains elements of both the TUSP and the RHP. From now on, we will refer to our problem as the Train Unit Shunting Problem with Reallocation (TUSP-R). Although a good working heuristic as developed by Van den Broek [3] to solve the TUSP-R is currently being evaluated and used by *NedTrain*, there exists a need for an exact, mathematically justified, and scientifically supported foundation. The overall goal of this research is to formulate the TUSP-R as an MIP and use mathematical programming techniques to solve it. We will compare our method to the currently used methods in order to evaluate its performance and evaluate its possible use for planning purposes.

The remainder of this thesis is structured as follows. In chapter 2 relevant literature on the problem and related problems is discussed and reviewed. Chapter 3 gives a more detailed description of the problem and introduces definitions and notations. The complexity of the problem is discussed in this chapter as well. Next, we formulate the problem as an MIP in chapter 4. We start with a basic formulation and work from there. Several extensions to the proposed formulation are provided as well. The solution approach is discussed in chapter 5 and the data sets we will apply it to are introduced in chapter 6. Chapter 7 discusses the results of our methods and a discussion of possible limitations of our model and methods is included in chapter 8. Our conclusion and a discussion of possible further research can be found in section 9.

Chapter 2

Literature Review

In this chapter we will provide an overview on relevant literature for this thesis. In the first section literature on several motion-planning problems including the RHP is discussed and their complexity is considered. In the second section we review works on general shunting problems and the third section is focused more specifically on the TUSP.

2.1 Motion-Planning Puzzles and their Complexity

The work of Flake and Baum [8] provides the foundation for a lot of articles on complexity of mathematical games, sliding block puzzles in particular. They elaborate on the complexity of the Rush Hour game, proving this to be $\mathcal{PSPACE}$-complete. Van Rijn [24] provides some extra aspects and clarifications of this proof. Flake and Baum [8] consider their main contribution to be their proof technique, which uses a lazy form of dual-rail reversible logic, requiring only three types of devices, which they claim makes it easily generalizable for other types of problems. Indeed, Hearn and Demaine [17] construct a framework which is a generalization of [8]. This framework is used to prove $\mathcal{PSPACE}$-completeness of several motion-planning problems. Hearn continues to work on complexity theory of different types of sliding-block and plank puzzles in [16]. He provides clear examples of such puzzles and elaborates on their complexity (which is thus $\mathcal{PSPACE}$-complete).

DePuy and Taylor [5] discuss several board puzzles which they claim encourage creativity in operations research students. One of the puzzles they discuss is Rush Hour. The authors formulate the problem as a mathematical programming problem. They use variables for the representation of vehicle moves, and variables for the board status. The objective function minimizes the number of moves needed to free the way to the exit for the target car. To keep the problem somewhat scalable, a maximum number of moves is allowed. We will also allow a maximum number of reallocations. They find that a maximum number of moves of four times the number of vehicles is reasonable. As for the computational results, they solve a simple instance in a computation time of only seconds using CPLEX 9.0.2.

2.2 General Shunting Problems

Blasum et al. [2] wrote one of the first works on depot scheduling problems. They focus on parking and dispatching trams in a depot and prove $\mathcal{NP}$-completeness for determining whether trams can be assigned to departure times without any shunting movements. This problem is also defined as the Tram Dispatching Problem. In their Tram Dispatching Problem, the depot consists only of LIFO tracks. The authors focus on a subproblem of the Tram Dispatching Problem, namely that of assigning already stored trams to departing services, while minimizing the number of necessary shunt movements. This subproblem is defined as the Scheduling-Trams-in-the-Morning Problem. In the Scheduling-Trams-in-the-Morning Problem it is assumed that operators do not take into account the departing services the next morning when assigning arriving trams to depot stacks. The latter is consistent with the current situation at transport companies, and it is also motivated

by the wish for real time planning: trams can very well arrive in a different order than scheduled due to delays, upsetting the entire planning.

In [25], Winter and Zimmermann elaborate on real-time dispatch problems in tram storage yards. Again, the storage yard or depot consists of LIFO tracks only. They define several dispatch problems. Among others, they consider the problem of minimizing the number of pairs of trams (both arriving and departing) that need to be shunted, the Minimum Shunting Problem. They also consider the Minimum Shunting at Departure Problem, which is identical to the Scheduling-Trams-in-the-Morning Problem in [2], meaning this is the Minimum Shunting Problem without considering the departing services while assigning arriving trams to stacks. The other problems they consider allow for type mismatches, which we do not allow for in this thesis, so these are of little relevance for us. The Minimum Shunting Problem is very relevant when formulating the complete TUSP-R, and the Minimum Shunting at Departure Problem is particularly relevant in case we would consider real time planning. $\mathcal{NP}$-completeness for all considered problems is proven. For the Minimum Shunting Problem the authors propose a binary problem and they model the Minimum Shunting at Departure Problem as a binary quadratic assignment problem. Considering several theorems, several constraints are added to the Minimum Shunting Problem resulting in a tightened quadratic assignment problem and this problem is linearized using a method by Kaufman and Broeckx [18]. This problem is solved exactly using the CPLEX 6.5 MIP-Solver. The Minimum Shunting at Departure Problem can be solved exactly in a similar manner. For both problems, some heuristic solutions are proposed as well.

In both [2] and [25], parking spaces are of fixed and identical size, which is impractical in the case there are vehicles of different lengths. Gallo and Di Miele [10] deal with city buses of different lengths that need to be parked in a parking depot. This depot consists of First In First Out stacks. The authors use the solution approach proposed in [25], meaning the problem is viewed as a quadratic assignment problem with side constraints. They proceed to extend this model to take into account buses with different lengths, thus the assumption of fixed parking spaces is abandoned. The problem is solved using a Lagrangean Decomposition approach as developed by Guignard and Kim [12].

2.3 Train Unit Shunting Problem

From 2002 on, multiple articles on the TUSP have appeared in literature. In [9] the TUSP is split up in two subproblems, namely the Matching Problem and the Track Assignment Problem. The Matching Problem entails matching the arriving and departing shunt units and the Track Assignment Problem involves parking these on a shunt track. The solution is a shunt plan, of which the costs are minimized. The problem is not treated as a whole, but as separate subproblems. The Matching Problem is formulated as a mathematical model and solved using the standard MIP solver of CPLEX 6.5. The Track Assignment Problem is also formulated as a mathematical model, using a set partitioning formulation with side constrains. In this formulation, all feasible track configurations are translated into a set, which the authors claim to be a major advantage since difficult constraints with respect to feasibility are avoided; they are incorporated into the set of feasible track configurations. The downside of this approach is the exponentially large set of track configurations. This downside is overcome by using a column generation heuristic. In this heuristic columns are generated using dynamic programming. Their approach does not allow for reallocation of parked shunt units and includes the matching problem. Also, routing is not included. In [21], a four-step solution approach is proposed to solve the TUSP. The first step is again matching arriving to departing shunt units, whereafter the routing costs of train units are estimated. Next, the shunt units are parked on shunt tracks and finally the actual routing is done. The Matching Problem is solved in a similar manner as in [9], and the Track Assignment Problem is also very similar. They do, however, include routing of shunt units, where [9] fails to do so. The Routing Problem is solved using the Occupied Network A* search algorithm extended with a 2-opt improvement strategy.

Di Stefano and Koči [6] focus on the subproblem in which they try to arrange the shunting units on the available tracks in such a manner that no shunting operations on outgoing trains are necessary in the morning. They use a graph theoretical approach in which they construct several algorithms and heuristics and do not use a mathematical program to formulate their problem. Their main goal is to investigate the constraints that make the shunting problem hard, and to that end they ignore constraints related to for example track lengths and shunt unit sizes. The constraints they say make the shunting problem hard are constraints that handle the shunting yard lay-out and the arrival and departure sequences of trains due to the timetable.

Di Stefano also worked on the Track Assignment Problem with Cornelsen [4]. To solve the problem, they use a conflict graph. In their conflict graph, the vertices correspond to the train units. Two train units are adjacent if they cannot be put on the same track. They show that the Track Assignment Problem is equivalent to coloring the conflict graph. In this case, the colors correspond to different tracks. For linear or aperiodic timetables, which are the ones discussed in this thesis, it is shown that the conflict graph is a permutation graph which hence can be colored in $\mathcal{O}(n \log n)$ time if the midnight constraint is respected. The midnight constraint states that arrivals and departures cannot be mixed in time.

Where [9] uses multiple subproblems to solve the TUSP in several steps, [19] and [20] solve the matching and parking subproblem simultaneously. They aim to include all possible extensions and configurations of trains and shunt yards, including trains consisting of multiple shunt units and both LIFO and free tracks. In their integrated approach, the authors aim to minimize the number of shunt units that arrive in the same train, but depart in different trains, meaning they need to be split up, as well as the number of tracks with multiple subtypes of train units parked on it. The complexity of their formulation is increased step by step in the paper, with only the possibility of reallocation missing compared to the formulation we aim to construct in this thesis. [20] lays out the basis for OPG, the planning system currently used by *NedTrain*. OPG is a MIP that decides on the parking and matching for each unit. OPG decides on which train units are assigned to which tracks and in which compositions they will leave the shunting yard again. Routing is added in the second step, using the TIMEFIXER method. TIMEFIXER is concerned with the process between entering the station and entering the park track.

In their 2007 article on circulation of railway rolling stock, Peeters and Kroon [23] develop a branch-and-price algorithm to assign trains to the daily train services. Their problem is not projected on a shunting yard, but the methods and formulations used can be useful for this thesis. They use a transition graph to build the mathematical formulation, on which they apply Dantzig-Wolfe decomposition. In this reformulation, a variable is associated with each path in the previously defined transition graph. Since this results in a huge number of path variables, a linear programming relaxation of the problem is solved using column generation. Subsequently, a branch-and-price approach is used to obtain an optimal integer solution. Fioole et al. [7] extend this model to incorporate the possibility of splitting and combining trains underway.

Den Hartog, in his 2010 thesis on shunt planning [15], discusses OPG and builds on the work of Lentink [20]. After analyzing this method, he concludes that the disadvantages of the sequential approach (that is, first applying OPG and subsequently applying TIMEFIXER) can only be fully overcome by integrating the two steps. To this end, the author formulates the sets Arrival on Park Track (APT) and Departure on Park Track (DPT). These sets are essential in formulating his APT model, an MIP that relies heavily on these sets. The set APT contains for all units all feasible, unique and full descriptions of the way a unit is handled between its arrival and its arrival at the park track. The set DPT is defined similarly: it contains descriptions for all units for the time between its departure from the park track and its overall departure from the shunting yard. Two methods are suggested for solving the APT model: a one- and a two-stage solution method. Since the APT and DPT sets are very large, they are only partially created to begin with in both solution methods. These initial sets are named start sets. Compared to the full sets, the start sets do not have any shunt time flexibility. The start sets contain for every composition and park track only the APT and DPT instances with the smallest dwelling time on the platform track. The

two-stage solution method starts with the same sets, but the flexibility in shunt time is added for some compositions in the second stage. The formulation and solution methods proposed in this thesis are interesting to use as inspiration for our formulation, since this thesis tackles the TUSP with all possible shunting yard lay outs incorporated, which we aim to do as well. Other papers often focus on one specific type of shunting yard lay out.

In [14] multiple methods for solving the TUSP are reviewed, developed, and compared. The authors introduce three new methods for solving the TUSP: they use a constraint programming formulation, develop a column generation approach, and construct a randomized greedy construction heuristic. They compare these methods to the existing methods using problem decomposition and the MIP formulation. In the latter, the possibility of delayed constraint generation is considered to decrease its memory consumption. The MIP formulation is very much based on the work of Kroon et al. [19]. The development of the constraint programming method is motivated by the nature of the TUSP, namely that it is mainly a feasibility problem. Not events, but compositions of multiple shunt units are assigned to tracks. The main sets used are the set of possible compositions of shunt units on tracks and the set of possible composition changes when an event happens. One can imagine these sets become extremely large, especially if we do not work with a preassigned matching of arriving to departing shunt units. This makes the memory consumption when solving the model with a constraint programming solver excessive. To prevent this, a heuristic variant is introduced in which the number of different train types allowed on one track is restricted. The column generation method decomposes the problem by track, and each track is assigned to a matching pattern. A matching pattern is a subset of matchings that can be feasibly parked on a given track over the planning horizon. This method is based on the work of Freling et al. [9], but the matching and parking subproblems are solved simultaneously. Column generation is applied to the problem, but since both matching and parking subproblems are solved at the same time, very large networks are constructed in the pricing problem resulting in large computation times. Therefore, the authors choose to not evaluate this approach thoroughly in the result section. The randomized greedy construction heuristic is not as interesting for this thesis. The paper only focuses on finding a feasible solution; no objective functions are taken into account. Besides this, only LIFO tracks are considered. Haahr, Lusby, and Wagenaar [14] also introduce infeasibility checks. If a problem instance fails the infeasibility checks, there is no reason to consider the instance at all. For example, if the aggregated track capacity is less than the sum of the train lengths that need to be parked at any moment in time, the instance will always be infeasible. This concept can help us to decrease computation times.

In another article by Haahr et al. [13], multiple methods for solving the Train Unit Parking Problem are considered. The Train Unit Parking Problem is similar to the TUSP, but the matching part is left out. They model the Train Unit Parking Problem using both a compact formulation focused on events, similar to the MIP formulations in [19] and [21], and a column generation model, also proposed in [9], obtained applying Dantzig-Wolfe Decomposition to the compact formulation. The first formulation is solved using a direct approach using the Gurobi solver, while they also test an approach in which they use a relaxed version of the model in which some constraints are removed. These constraints are gradually reintroduced using a Branch-and-Cut procedure. This second approach is motivated by the fact that there is a very large number of constraints in the compact formulation, which the authors claim might not all need to be generated from the start. We will keep this idea in mind when developing our solution methods. The second model is characterized by exponentially many variables, which motivates the choice for using the column generation model as proposed by [9], extended to a full Branch-and-Price framework through constraint branching. They finally propose a unit swapping algorithm in case not all units can be parked in a solution to the Train Unit Parking Problem and a framework that solves the Train Unit Parking and Matching Problem in an integrated loop, effectively solving the TUSP. They conclude that a direct solve of the full MIP gives the best results, rendering the more complicated column generation approach unnecessary, although they state it would be interesting to see which method would scale better for larger instances. This is an interesting conclusion to keep in mind in this thesis.

Finally, Van den Broek [3] focused on constructing a simulated annealing algorithm for finding

shunt plans for the TUSP with Service Activities. The shunt plans include the matching, the scheduling of the service tasks, the assignment of trains to parking tracks, and the routing of trains. He developed an integrated approach that evaluates all of the components of the shunt plans simultaneously rather than subsequently. The performance of the heuristic is considerably better than the OPG-method. His method is currently being used by *NedTrain* for determining service site capacities. His work will be very useful for measuring the effectiveness and evaluating the results of our methods, since he also generated results for the TUSP without Service Activities, since our models will solve the matching, parking, and routing subproblems simultaneously as well. However, we will aim to construct an exact approach, where Van den Broek [3] proposed a heuristic one.

Chapter 3

Problem description

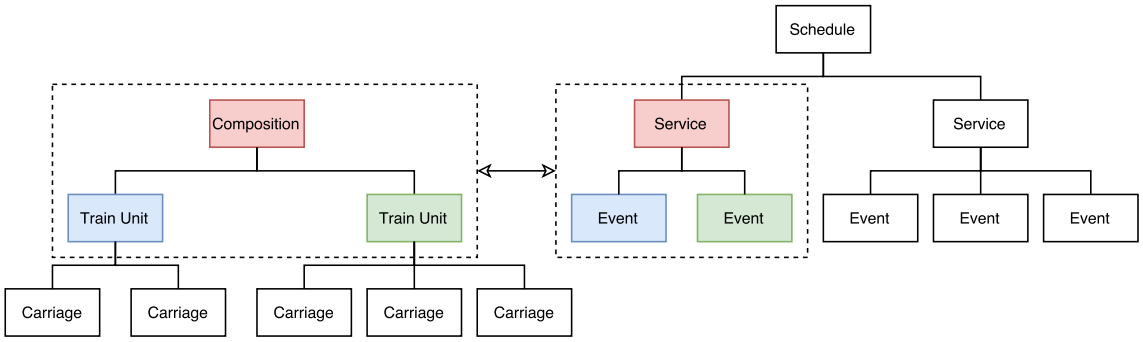
This chapter will provide a formal problem description of the TUSP-R. The general terminology will be introduced and we will provide the general notation for the sets, parameters, and (decision) variables used later on. We will also comment on the time complexity of the problem.

3.1 General terminology

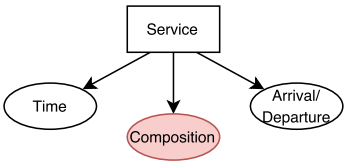
We need to introduce a clear definition of concepts that will be used throughout this paper to clearly define the problem. First of all, we define a shunting yard as the place where train units are parked when they are not needed for (passenger) transportation on the main railway network and where maintenance activities can be performed on them. For each shunting yard, we define an A- and a B-side. For a horizontally orientated shunting yard, the A-side usually refers to the left side of the shunting yard, while the B-side refers to the right side of the yard. The A- and B-sides of tracks are the sides that are closest to the A- or B-sides of the shunting yard, respectively. Shunting yards can have different layouts and makeups, with LIFO and free tracks. LIFO tracks are tracks that can only be accessed from either the A- or B-side; free tracks can be accessed from both sides. As one can imagine, the shunting yard layout has considerable implications for the used parking strategies.

On a given day, train units arrive at and depart from a shunting yard from and to the main rail network at pre-determined moments in time in pre-defined compositions.Compositions are multiple train units coupled together, but they can also consist of only one train unit. Compositions can only consist of train units with the same type. However, these train units can have different numbers of carriages. Train units are the smallest indivisible units of which compositions can consist and they can drive independently of one another. Carriages are the separate compartments in a train unit. We refer to a composition with a pre-defined makeup of train unit type and numbers of carriages arriving at or departing from the shunting yard according to a certain time schedule as a train service. Arrivals of train units at the shunting yard and departures of train units from the shunting yard are referred to as events. Events are train unit based: if a service consists of multiple train units, a separate event is defined for each train unit in this service. However, each train unit has its own position within this service, for example the front, back, or middle. An overview of these definitions is given in figure 3.1.

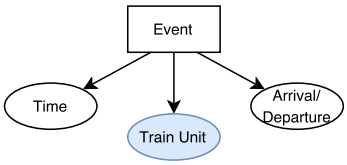
The right side of figure 3.1a represents a schedule, which consists of services, that on their turn consist of one or more events. Services have three characteristics (see figure 3.1b): whether they are arrivals or departures, their time stamp, and the composition they are linked to. As can be seen in figure 3.1c, events have similar characteristics: whether they are arrivals or departures, their time stamp, and the train unit they are linked to. Compositions consist of one or more train units, which consist of multiple carriages. In this case, the schedule consists of two services, one consisting of two events and the other consisting of three events. The composition linked to the former thus consists of two train units, of which one contains two carriages, and the other contains three carriages. There are certain restrictions to the follow up times of different train services in a schedule: there needs to be a window of at least 4 minutes between two train services that move



(a) Make up of schedules and compositions



(b) Characteristics of a train service



(c) Characteristics of an event

Figure 3.1: Relations schedules, services, events, compositions, train units, and carriages

in the same directions (that is between two arrivals or between two departures) and of at least 7 minutes between two train services that move in opposite directions (that is between an arrival and departure or vice versa).

In this thesis, the aim is to find a way to feasibly park train units arriving in the train services in the shunting yard such that they can depart in suitable departing train services. We assume no train units are present in the shunting yard at both the start and the end of the planning period, which means that all train units arrive and depart in a train service. This also means that the train unit types and the number of carriages should be of equal numbers in the arriving and departing compositions, otherwise train units would have to stay behind in the shunting yard or there would be shortages of train units in the departing services. Seeing that all train units arrive in an arriving service and depart in a departing service, some sort of link or connection must be made to assign arriving to departing units and vice versa. We call this a matching: an overview of the train services a train unit arrives and departs in. Each train unit in an arriving service is matched to a train unit in a departing service, and the other way around. Train units can only be matched if the type and the number of carriages are equal. Also, a matching between an arriving train unit and a departing train unit can obviously only be made when the arrival takes place before the departure. For some train units, a predefined matching exists, which means we can not freely match the unit to a service in which a unit with the same type and number of carriages is required. That is, only one specific matching can be made for these units. This can occur when we want a particular train unit to be at a specific location at a given day, for example to execute a specialized maintenance service or update. If this is the case, the train unit is not interchangeable. Otherwise, the train unit is interchangeable. Compositions consisting of multiple train units can be split up into smaller compositions and train units; compositions can also be 'merged' to together form a composition of even more train units. The former is referred to as decoupling, the latter as coupling. The need to couple and decouple train units exists for example when the train units in the composition of a specific arriving service do not all depart (in the same order) in the same departing service or when the arriving composition forms only part of the departing composition.

During their stay in the shunting yard, train units need to be parked on one of the parking tracks. Parking tracks are tracks on which we can park train units, and they are usually connected with the main railway network through (a) diagonal track(s). We do not park any train units on diagonal tracks. Train units are parked on a parking track directly after their arrival in the shunting yard. This track is referred to as the arrival track. We call the track on which they are

parked right before their departure from the shunting yard the departure track. A train unit can be parked on multiple tracks during its stay in the shunting yard and the arrival and departure track can be different. The arrival and departure track are part of the main railway network. The arrival and departure track are both part of the shunting yard, and not of the main railway network. When a train unit moves from one parking track to another during its stay in the shunting yard, we say the train unit reallocates. If a train unit does not reallocate during its stay in the shunting yard, it occupies only one track during its stay in the yard, which is both the arrival and departure track. If it reallocates once during its stay in the shunting yard, the parking tracks it occupies are the arrival and departure track. If it reallocates multiple times, it first occupies the arrival track, then some track(s) in between, and finally the departure track. Reallocation and the other definitions and concepts discussed in this paragraph have already been introduced in section 1 by means of table 1.1 and figure 1.5. In the figure, the thin lines form the shunting yard and the diagonal track on the right is the connecting track on which we do not park train units. The horizontal tracks with train units parked on them are parking tracks. When elaborating on the example illustrated in table 1.1 and figure 1.5, the concept of drive back distance was also introduced: the drive back distance is the maximum length a train unit can have to feasibly move between two parking tracks without using the main railway network. For a shunting yard layout such as the one in figure 1.5, the drive back distance to move from one parking track to another is equal to the length of the diagonal track from the highest parking tracks of the two to the main railway network. Reallocations are useful to avoid crossings. Crossings occur when a train unit would have to move through another unit in order to be able to leave the shunting yard in its train service. The train unit that is preventing free passage is said to be blocking the departing train unit. Crossings are obviously not allowed. If reallocation is not allowed, train service 6 would not be able to depart in time, since train service 3 (7) would be blocking the way, which would mean a crossing would occur. This would also be the case if reallocation were allowed but the drive back distance would not be sufficient to reallocate train service 3 (7). For LIFO tracks, crossings occur if a train unit arrives on a certain track before another train unit, and this first train unit also departs from this track before the second one, but after the arrival of the second train unit on the track.

We can now formulate the problem as follows:

Given

- *a shunting yard;*
- *and a train service schedule;*

we need to

- *assign arriving and departing train units to tracks;*
- *match arriving and departing train units to each other;*
- *choose reallocation times for the matchings;*
- *and route the train units over the tracks;*

such that

- *at no point in time the cumulative length of train units on a track exceeds the track length;*
- *no crossings occur at either reallocation or final departure of train units from the shunting yard;*
- *the main railway network is not used in the process of reallocating train units;*
- *and no simultaneous movements of train units take place;*

while minimizing the number of reallocations.

Although we aim to minimize the number of reallocations, it is mainly a feasibility problem. Note that previously defined linear programming formulations for the TUSP as discussed in chapter 2 generally implicitly include the routing element: they use shunting yard layouts in which train units can easily reach all tracks when arriving in the shunting yard. To be able to feasibly reallocate between tracks, we need some extra constraints to force correct routing. Extra restrictions on routing apply as well when we consider more complicated shunting yard layouts.

Set	Definition
F	Set of tracks
T	Set of all train units/events
$T_+ \subset T$	Set of arriving train units/events
$T_- \subset T$	Set of departing train units/events
L	Set of all possible matches between arriving and departing train units

Table 3.1: Set overview

Parameter	Definition
k_t	Length of train unit $t \in T$
l_f	Effective track length of track $f \in F$
$n_{f,g}$	Maximum length of a train composition to be able to go from track f to track g, $f, g \in F$
τ_t	Type of train unit $t \in T$
$v_t =$	$\begin{cases} 1 & \text{if train unit } t \in T \text{ is an arriving train unit, that is } t \in T_+ \\ -1 & \text{otherwise}^1 \end{cases}$
ω	Fixed moving time from one location to another in the shunting yard
w_t	The number of reallocations that can take place between event t and $t-1$, $t \in T$, that is the time between event t and $t-1$ in terms of the moving time ω minus 1 if $w_t > 0$

Table 3.2: Parameter overview

3.2 General notation

In order to mathematically formulate the problem, we introduce some notation on amongst others the concepts discussed in section 3.1. First of all, we define the set of tracks F, with parameter l_f the length of track $f \in F$. T_+ is the set of arriving train units or events, while T_- is the set of departing train units or events. Together, sets T_+ and T_- form set T and they are disjoint. In mathematical notation, we have:

$$T = T_+ \cup T_-,\ T_+ \cap T_- = \emptyset.$$

The parameter k_t gives us the length of each train unit $t \in T$ and τ_t assigns the type to each unit $t \in T$. We define a unique type τ_t for each existing combination of train unit type and number of carriages: if two train units are both of the train type SLT, but one has 4 carriages and the other 6, their τ_t value is different. We also use parameter v_t, which is equal to 1 if unit t is an arriving train unit, i.e. $t \in T_+$ and equal to -1 if unit t is a departing train unit, i.e. $t \in T_-$.

The elements in set T do not only refer to train units, but also to their associated events and thus the event times. Therefore we can sort the sets T, T_-, and T_+. This ordering is taken from [19] and we order the elements of these sets based on event time and position in the service the train unit belongs to. A train unit t_1 appears before train unit t_2 in the ordering (denoted by $t_1 < t_2$) if and only if one of the following conditions is satisfied:

1. The event time of train unit t_1 is smaller than the event time of train unit t_2;
2. Train units t_1 and t_2 are arriving train units that arrive in the same train service, and train unit t_1 is positioned more to the front of the composition than train unit t_2;
3. Train units t_1 and t_2 are departing train units that depart in the same train service, and train unit t_1 is positioned more to the front of the composition than train unit t_2.

Because of the restrictions on the follow up times between services in a schedule as discussed in section 3.1, we do not need to consider cases where we have simultaneous events that do not belong to the same service. Therefore, we will for example never encounter cases where the event time of

[1] I.e. if train unit $t \in T$ is a departing train unit, that is $t \in T_-$

an arriving train unit is equal to the event time of a departing train unit.

We can now define w_t as the time between event t and $t-1$, $t \in T$, in terms of the moving time, minus 1 if w_t is larger than 0, meaning that w_t is equal to the number of reallocations that can take place between event t and $t-1$. We consider the moving time a fixed constant ω, meaning it is independent from the distance that needs to be covered. The drive back distance between two tracks is defined by $n_{f,g}$, which means that $n_{f,g}$ is the maximum length of a train composition to be able to go from track $f \in F$ to track $g \in F$, set to $M > \max_{t \in T}\{k_t\}$ if there is no limit on this length.

Next, the set L contains all possible matches (t, u) between arriving train units $t \in T_+$ and departing train units $u \in T_-$. Since matches can only made if the arriving train unit was already present in the shunting yard before the departing service departs, we can formulate the set L as follows:

$$L = \{(t, u) | t \in T_+, u \in T_-, t < u, \tau_t = \tau_u\}.$$

In table 3.1 and table 3.2, an overview of the used sets and parameters is given.

3.3 Complexity

The complexity of the TUSP and similar problems has been discussed in several articles appearing in literature. Blasum et al. [2] and Winter and Zimmerman [25] prove for several of their tram dispatching problems that they are $\mathcal{NP}$-hard, which Freling et al. [9] consider sufficient proof for their TUSP to be $\mathcal{NP}$-hard. Haahr et al. [14] consider the TUSP a feasibility problem and they prove $\mathcal{NP}$-hardness of their TUSP by reduction from the Graph Coloring problem.

The TUSP-R is an $\mathcal{NP}$-hard problem, since it can be verified in polynomial time whether a certain assignment of values to variables (an instance) is a solution to the problem and by comparison to the TUSP. We can verify the matching constraints by counting the number of assignments. We ensure that each arriving (departing) train unit is matched to exactly one departing (arriving) train unit and that the assignment is mutual. We can verify the capacity constraints by iterating over the ordered events and updating the cumulative lengths of the train units on the tracks and the number of movements in a time interval. These should be within the allowed limits. Each event should be assigned to a track as well. By pairwise comparison of the matchings in combination with their track assignments and reallocation times, we can ensure no crossings occur. Thus we can verify in polynomial time whether an instance is a solution. Furthermore, if we find the optimal solution to the TUSP-R and in this solution zero reallocations occur, the solution to the TUSP-R is a solution to the TUSP as well. If we find an optimal solution to the TUSP-R with multiple reallocations in it, no solution will exist for the TUSP- Therefore, if the TUSP-R would be solvable in polynomial time, the TUSP would be as well. However, the TUSP is an $\mathcal{NP}$-hard problem as shown in [9] and [14] and therefore the TUSP-R is as well.

Chapter 4

MIP Formulation

In order to work towards the most efficient formulation for our problem, we start with a basic formulation and work from there. In this initial formulation in section 4.1, we use variables with a lot of indices, which we expect will not be the most practical. However, this will give us a feel for the problem dynamics. Later on in section 4.2 we adapt the choice of variables to find a more appropriate formulation.

We assume all train services consist of one train unit only. That means no couple and/or decouple actions need to be performed. Although we will provide extensions to dismiss this restriction in section 4.3, we will still work with a simplified model: we will work under the assumption that compositions consisting of multiple units are decoupled directly after their arrival in the shunting yard and coupled right before their departure. This means all models will operate on the train unit level. Also, we use a track layout with LIFO tracks, that are all on the same side of one track that connects them to the main rail network. An example of such a layout can be found figure 4.1. In section 4.3 adaptions will be provided to make the model suitable for all shunting yard layouts. In our models, only one train unit can be moved at the same time and the operation of switches is not considered.

Our starting point for finding a formulation for the TUSP-R is the assumption that reallocation is given in by a train unit trying to leave a track, but being unable to do so since other train units have arrived later on the LIFO track and have not left the track yet. In this case, the latter train unit will need to reallocate. From this point of view, we build our formulation. We will allow at most one reallocation per train unit in our MIP formulation. Allowing for more reallocations will greatly complicate solving the model exactly, as the number of variables and constraints will both increase roughly by a factor n each time, where n is the number of extra reallocations allowed. Since the dimensions of the constraint matrix are already quite big as can be seen in the formulations in this chapter, the computation time is expected to increase drastically. Besides this, we do not expect allowing for multiple reallocations will greatly reduce the number of unsolvable cases, unless we deal with very uncommon sorts of shunting yard layouts with very few, but very long tracks. We do not expect our solution to improve a lot when allowing for multiple reallocations per train unit, since reallocations take time and we expect the marginal benefit of reallocations

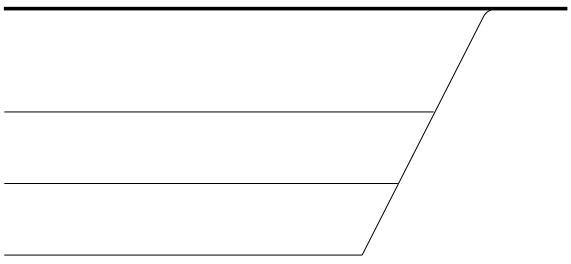


Figure 4.1: Example track layout only LIFO tracks

decreases rapidly with each extra reallocation allowed.

We assign a time window to reallocations in the formulations in this section, where between each event and its successor event a time window exists. Since we only allow one train unit to move at a time in the shunting yard, time windows cannot span multiple events, as a train unit will always be moving during an arrival at or departure from the shunting yard. For the same reason, we limit the number of reallocations that can be assigned to a time window to ω_t as discussed in chapter 3. However, we do not determine the order of reallocations within a time window. We assume we will always be able to reallocate during the time window such that no crossings occur. Furthermore, we assume that when departing from the shunting yard, the reallocation order of train units that have used the same reallocation time window as the departing unit can always be chosen such that no crossings involving these train units will occur. These assumptions simplify the model and reduce the number of variables used, but do not always hold. In the situation where we reallocate two train units from the same arrival track to the same departure track during the same reallocation time window, the train unit that arrived last on the arrival track must be reallocated first. We will refer to this train unit as train unit A. The train unit that is reallocated second is referred to as train unit B. Now, train unit A will arrive at the departure track before train unit B. The first assumption holds, because we just found a reallocation order in which no crossings occur at reallocation. The second assumption, however, does not necessarily hold: if train unit A needs to depart from the shunting yard before train unit B, a crossing will occur. If we would have reversed the order of reallocation of units A and B, no crossing would occur, but since their arrival tracks are identical, this is not possible. So when dealing with identical reallocation times, arrival tracks, and departure tracks, our assumption might not hold. We do not expect this situation to arise often.

We found the biggest challenge in formulating our problem as an MIP to be formulating the constraints that prohibit crossings. Since we have to deal with a variable matching and a variable reallocation time, it is difficult to determine the exact time period during which a train unit is present on a certain track, which is vital information when dealing with crossings. For two matchings (t, u) and (t', u'), $(t, u), (t', u') \in L$, crossings occur if one matching arrives on a certain track before the other one, and this first matching also departs from this track before the other one. We use several different variables in the sections of this chapter to formulate our problem as smartly as we can. An overview of all the used variables per section is given in appendix A.

4.1 Initial formulation

In this section we will define an introductory notation. In the formulation, we assign arriving train units to tracks. When these train units need to depart the track for reallocation or final departure, train units that are blocking free passage can be moved. We minimize the number of reallocations, but this objective can easily be adapted to suit the planner's wishes.

In order to determine which train units will be matched to each other and where they will be parked throughout their stay in the shunting yard, we will need several decision variables. We use the variables $x_{t,f}$, $x_{t,u,f,g}$, $y_{t,u,f,g,z}$, and $b_{f,t}$. The x-variables are all concerned with the parking tracks of train units and their matchings, while the y-variables also incorporate the reallocation time.

$x_{t,f}$ is equal to 1 if train unit $t \in T$ is assigned to track $f \in F$, and 0 otherwise. Note that if we are dealing with an arriving train unit t ($v_t = 1$), the track assignment f will be the arrival track, while if we are dealing with a departing train unit t ($v_t = -1$), the track assignment f will be the departure track. $x_{t,u,f,g}$ equals 1 only if arriving train unit t assigned to track $f \in F$ is matched to departing train unit u assigned to track $g \in F$, $(t, u) \in L$. Next, $y_{t,u,f,g,z}$ is just like the previous ones a binary variable that is only equal to 1 if arriving train unit t assigned to track $f \in F$ is matched to departing train unit u assigned to track $g \in F$, $(t, u) \in L$, and reallocated between event $z - 1$ and z, $z \in T$. Finally, $b_{f,t}$ equals the cumulative length of the train units on track $f \in F$ right after event $t \in T$.

The above sets of variables account for a very large number of total variables. The formulation will contain $|T| \cdot |F|$ $x_{t,f}$-variables; $|L| \cdot |F|^2$ $x_{t,u,f,g}$-variables; approximately $|L| \cdot |F|^2 \cdot |T|$ $y_{t,u,f,g,z}$-variables; and $|F| \cdot |T|$ $b_{f,t}$-variables. Note that $|L|$ is in the order of magnitude of $(\frac{1}{2}|T|)^2 = \frac{1}{4}|T|^2$. Therefore, the number of columns in the constraint matrix will be in the order of magnitude of

$$2(|T| \cdot |F|) + (1 + |T|)(|L| \cdot |F|^2) \in \mathcal{O}(\frac{1}{4}|T|^3 \cdot |F|^2).$$

We can now formulate the problem as follows.

Minimize:

$$\sum_{(t,u)\in L} \sum_{f\in F} \sum_{g\in F} \sum_{\substack{z\in T:\\ z\neq t}} y_{t,u,f,g,z} \tag{4.1}$$

Subject to:

$$\sum_{f\in F} x_{t,f} = 1 \qquad \forall t \in T \tag{4.2}$$

$$\sum_{\substack{u:\\ (t,u)\in L}} \sum_{g\in F} x_{t,u,f,g} = x_{t,f} \qquad \forall t \in T_+,\ f \in F \tag{4.3}$$

$$\sum_{\substack{t:\\ (t,u)\in L}} \sum_{f\in F} x_{t,u,f,g} = x_{u,g} \qquad \forall u \in T_-,\ g \in F \tag{4.4}$$

$$\sum_{\substack{z\in T:\\ t<z\leq u}} y_{t,u,f,g,z} = x_{t,u,f,g} \qquad \forall f,g \in F : f \neq g,\ (t,u) \in L \tag{4.5}$$

$$y_{t,u,f,f,t} = x_{t,u,f,f} \qquad \forall f \in F,\ (t,u) \in L \tag{4.6}$$

$$\sum_{f\in F} y_{t,u,f,g,z} + \sum_{\substack{z'\in T:\\ u<z'\leq u'}} \sum_{g'\in F} y_{t',u',g,g',z'} \leq 1 \qquad \begin{array}{l}\forall (t,u) \in L,\ z \in T : t \leq z \leq u \\ (t',u') \in L : z \leq t' < u, u' > u,\ g \in F\end{array} \tag{4.7}$$

$$\sum_{f\in F} y_{t,u,f,g,z} + \sum_{\substack{z'\in T:\\ t'\leq z'\leq u,\\ z'>z}} \sum_{f'\in F} y_{t',u',f',g,z'} \leq 1 \qquad \begin{array}{l}\forall (t,u) \in L,\ z \in T : t \leq z \leq u \\ (t',u') \in L : t' < u, u' > u,\ g \in F\end{array} \tag{4.8}$$

$$\sum_{g\in F} y_{t,u,f,g,z} + \sum_{\substack{z'\in T:\\ z<z'\leq u}} \sum_{g'\in F} y_{t',u',f,g',z'} \leq 1 \qquad \begin{array}{l}\forall (t,u) \in L,\ z \in T : t < z \leq u \\ (t',u') \in L : t < t' < z, u' > z,\ f \in F\end{array} \tag{4.9}$$

$$\sum_{g\in F} y_{t,u,f,g,z} + \sum_{\substack{z'\in T:\\ t<z'<z}} \sum_{f'\in F} y_{t',u',f',f,z'} \leq 1 \qquad \begin{array}{l}\forall (t,u) \in L,\ z \in T : t < z \leq u \\ (t',u') \in L : t' < z, u' \geq z,\ f \in F\end{array} \tag{4.10}$$

$$b_{f,0} = 0 \qquad \forall f \in F \tag{4.11}$$

$$\begin{aligned} b_{f,t-1} + v_t k_t x_{t,f} + \sum_{\substack{(t',u')\in L\\ t'<t\leq u'}} \sum_{f'\in F\setminus f} y_{t',u',f',f,t} k_{t'} \\ - \sum_{\substack{(t',u')\in L\\ t'<t\leq u'}} \sum_{g'\in F\setminus f} y_{t',u',f,g',t} k_{t'} = b_{f,t} \end{aligned} \qquad \forall t \in T,\ f \in F \tag{4.12}$$

$$b_{f,t} \leq l_f \qquad \forall t \in T,\ f \in F \tag{4.13}$$

$$\sum_{\substack{u:\\ (t,u)\in L}} x_{t,u,f,g} k_t \leq n_{f,g} \qquad \forall f,g \in F,\ t \in T \tag{4.14}$$

$$\sum_{\substack{(t,u)\in L:\\ t<z\leq u}} \sum_{f\in F} \sum_{g\in F} y_{t,u,f,g,z} \leq w_z \qquad \forall z \in T \tag{4.15}$$

$$x_{t,f} \in \{0,1\} \qquad \forall t \in T,\ f \in F \tag{4.16}$$

$$x_{t,u,f,g} \in \{0,1\} \qquad \forall t,u \in T,\ f,g \in F \tag{4.17}$$

$$y_{t,u,f,g,z} \in \{0,1\} \qquad \forall t,u,z \in T,\ f,g \in F \tag{4.18}$$

$$b_{f,t} \geq 0 \qquad \forall f \in F,\ t \in T \tag{4.19}$$

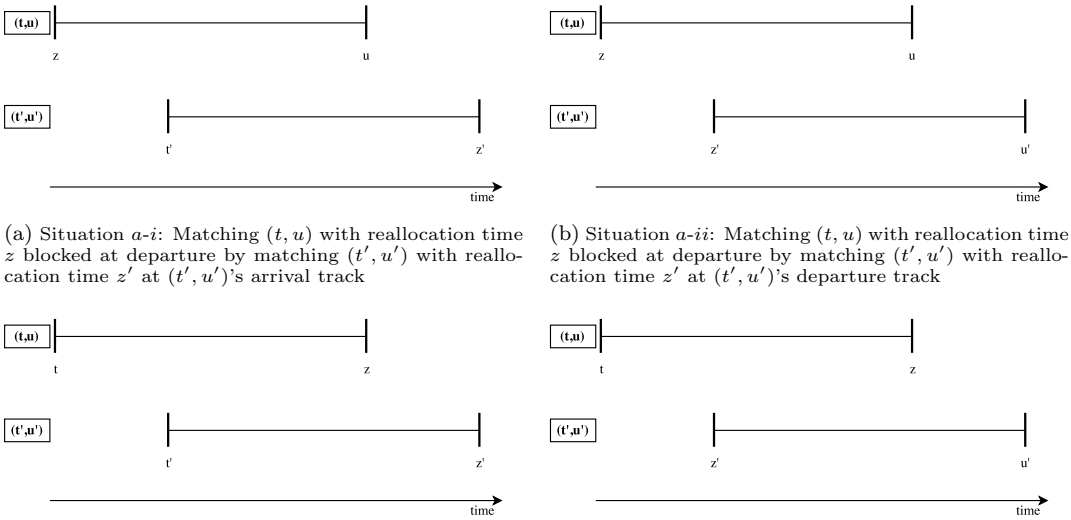
First of all, the objective function (4.1) minimizes the number of reallocations. In constraints (4.2), each event is assigned to a track. Constraints (4.3) and (4.4) ensure that each arriving train unit is matched to exactly one departing train unit and vice versa. In case $\sum_{f\in F} \sum_{\substack{g\in F:\\ g\neq f}} x_{t,u,f,g} = 1$ for a certain matching (t,u), which means the arrival track of unit $t \in T_+$ is not equal to the departing track of unit $u \in T_-$ and these train units are matched to each other, the train unit will have to move from track f to track g at a certain time during the planning period. Therefore, we introduce constraints (4.5). These constraints ensure the unit is actually moved at a certain point in time when this is necessary and help decide on the moment in time on which the reallocation takes place. If no reallocation needs to take place, we define the reallocation time to be equal to the arrival time, which is shown in constraints (4.6).

Next, constraints (4.7)-(4.10) ensure that no crossings occur. Crossings occur when a matching arrives at a track after the arrival of another matching at the track, but before the departure of this matching from the track, and departs from the track after the departure of the matching already present at the shunting yard. If we define $\alpha_{t,u,f}$ as the arrival time of matching (t,u) on track f, and we define $\delta_{t,u,f}$ as the departure time of matching (t,u) from track f, we can distinguish four situations for any combination of matchings (t,u) and (t',u') that occupy a track f simultaneously for any period of time in the planning period. These options are given in table 4.1. There is no need to consider both columns of the table in the future, since the second column illustrates the same situations as the first for the opposite choices of (t,u) and (t',u'). In short, we can say that a train unit uses two tracks when reallocating. It can be blocked by other train units when it departs from either of these tracks. In constraints (4.7) and (4.8), crossings at departure from the second track are prevented, which is the track a matching has been reallocated to and from which it departs to leave the shunting yard for good. We refer to this situation as situation a. In these constraints, we also prevent crossing for train units that are never reallocated. These train units only use one track during their stay in the shunting yard. Therefore, in these cases, they can only be blocked at their departure from the shunting yard. In constraints (4.9) and (4.10), crossings at departure from the first track are prevented, which is the track a matching is initially parked at and from which it reallocates to its final departure track. We refer to this situation as situation b. Train units can be blocked at their departure from a track by train units that use the same track as arrival track or departure track. If they use the same track as arrival track, they are reallocated away from this track at a certain point in time, in which case their stay on a track lasts from their arrival time to their reallocation time. If they use it as departure track, the blocking train units are reallocated to this track at a certain point in time, in which case their stay on a track lasts from their reallocation time to their departure time from the shunting yard. We refer to the former as situation i, and to the latter as situation ii. Situation i is prevented in constraints (4.7) and (4.9), while situation ii is prevented in constraints (4.8) and (4.10). If a train unit is not reallocated during its stay in the shunting yard, the reallocation time equals the arrival time, so this case is incorporated in the latter blocking possibility. In figure 4.2, an overview of the above is given by the means of four timelines. In this figure, a crossing occurs if the stated

	$\delta_{t,u,f} < \delta_{t',u',f}$	$\delta_{t,u,f} > \delta_{t',u',f}$
$\alpha_{t,u,f} < \alpha_{t',u',f}$	Crossing	No crossing
$\alpha_{t,u,f} > \alpha_{t',u',f}$	No crossing	Crossing

Table 4.1: Crossing overview LIFO track

matchings are made and they are parked at the same track in one of the illustrated situations.



(a) Situation *a-i*: Matching (t, u) with reallocation time z blocked at departure by matching (t', u') with reallocation time z' at (t', u')'s arrival track

(b) Situation *a-ii*: Matching (t, u) with reallocation time z blocked at departure by matching (t', u') with reallocation time z' at (t', u')'s departure track

(c) Situation *b-i*: Matching (t, u) with reallocation time z blocked at reallocation by matching (t', u') with reallocation time z' at (t', u')'s arrival track

(d) Situation *b-ii*: Matching (t, u) with reallocation time z blocked at reallocation by matching (t', u') with reallocation time z' at (t', u')'s departure track

Figure 4.2: Crossing possibilities LIFO tracks

Summarizing, in constraints (4.7) situations of type *a-i*, as illustrated in in figure 4.2a, are prevented. We sum over $f \in F$ in the first summation and over $g' \in F$ in the third summation, since the arrival and departure track of matchings (t, u) and (t', u'), respectively, are not relevant here. In constraints (4.8) situations of type *a-ii*, as illustrated in in figure 4.2b, are prevented. Here, we again sum over $f \in F$ in the first summation and we sum over $f' \in F$ in the third, since the arrival tracks of matchings (t, u) and (t', u') are not relevant here. In constraints (4.9) situations of type *b-i*, as illustrated in in figure 4.2c, are prevented. This time, we sum over $g \in F$ and $g' \in F$ in the first and third summation, respectively, since the departure tracks of both matching $(t, u) \in L$ and (t', u') are not relevant here. In constraints (4.10), situations of type *b-ii*, as illustrated in in figure 4.2d, are prevented. We sum over $g \in F$ in the first summation and we sum over $f' \in F$ in the third, since the departure track of matching (t, u) and the arrival track of matching $(t', u') \in L$ are not relevant here. In each of the constraints (4.7)-(4.10) we choose the combinations of (t, u), (t', u'), and z and sum over $z' \in T$ in the second summation in such a way that we are actually dealing with a potential crossing of the type we aim at in the particular constraints.

The cumulative length of the train units occupying a certain track at a certain point in time is updated and regulated by constraints (4.11) and (4.12). Since constraints (4.12) use a recursive relation by means of variable $b_{f,t-1}$, we need to initialize $b_{f,0}$ for all $f \in F$ in constraints (4.11). Constraints (4.12) then update the occupied track lengths by adding (subtracting) the length of the current arriving (departing) train unit if it is assigned to the track under consideration and adding (subtracting) the length of all train units that are reallocated to (from) the track right before the current event. To ensure this cumulative length does not surpass the effective track length, we introduce constraints (4.13). Constraints (4.14) make sure a unit can actually reach the track they need to be reallocated to, while constraints (4.15) limit the number of reallocations in a time period to the maximum possible number of reallocations in this time period: the maximum number of reallocations right before an event is limited to w_z. Finally, constraints (4.16)-(4.19) define the domain of the used variables.

The number of constraints in the formulation discussed in this section is in the order of mag-

nitude of

$$2|T| + |F| + 3(|T| \cdot |F|) + |F|^2(|T| + |L|) + |F| \cdot |L| + 4(|L|^2 \cdot |T| \cdot |F|) \in \mathcal{O}(\frac{1}{4}|T|^5 \cdot |F|),$$

which is approximately the number of rows in the constraint matrix, excluding domain restrictions. Therefore, the dimensions of the constraint matrix will be in the order of magnitude of

$$\big[2(|T| \cdot |F|) + (1 + |T|)(|L| \cdot |F|^2)\big]$$
$$\times$$
$$\big[2|T| + |F| + 3(|T| \cdot |F|) + |F|^2(|T| + |L|) + |F| \cdot |L| + 4(|L|^2 \cdot |T| \cdot |F|)\big]$$

4.2 Final formulation

The model we described in section 4.1 is rather extensive, using variables with a lot of indices, which means the number of variables and constraints quickly grows as the number of train units and tracks considered becomes larger. When reconsidering, we see that we may not always need all the information enclosed in the indices, which could lead to a smaller model and less complicated constraints.

In the objective function (4.1), for example, we do not need information on the tracks that train units reallocate between, or in which matching they belong. Also, we see that the information enclosed in the indices g and f in $x_{t,u,f,g}$ if we take a look at constraints (4.2) and (4.3), respectively, is superfluous. In these constraints, no relevant information is enclosed in them and they cause the summations in these constraints to be unnecessarily extensive. Similarly, in constraints (4.7)-(4.10), we want to ensure that no crossings occur. To this end, we need information on the arrival and departure times of train units on the considered track. It is, however, not relevant in this case where they originate from or where they leave to - hence the summations over the tracks. Comparable reasoning can be done for constraints (4.12). In constraints (4.14), we only need information on between which tracks a train unit reallocates and timing is not important, and contrarily in constraints (4.15) we are only interested in the timing of a reallocation.

To overcome the issues of using large amounts of variables that contain too much detail, we propose a new formulation, again based on an integrated approach for parking and matching as described by Kroon et al. [19]. We try to find a balance between the number of variables and constraints and to keep both as small as we can. To this end, it is sometimes convenient to use the product of variables and other non-linear features. For practical solving purposes, however, we will provide methods to rewrite elements of the formulation in such a way that we have a linear formulation again in subsection 4.2.1.

Besides the already introduced variables $x_{t,f}$ and $b_{f,t}$, we will need to introduce some extra variables for this formulation. First of all, we have binary variables $x^1_{t,u,f}$ and $x^2_{t,u,f}$. $x^1_{t,u,f}$ is equal to 1 if and only if arriving train unit $t \in T_+$ is matched to departing train unit $u \in T_-$ and the arrival track is $f \in F$, which means $x_{t,f}$ is also equal to 1; $x^2_{t,u,f}$ is equal to 1 if and only if arriving train unit $t \in T_+$ is matched to departing train unit $u \in T_-$ and the departure track is $f \in F$, which means $x_{u,f}$ is also equal to 1. Next, we have $m_{t,u,z}$, again a binary variable, which is equal to 1 if arriving train unit $t \in T_+$ is matched to departing train unit $u \in T_-$ and is reallocated between event $z-1$ and z, and equal to 0 otherwise. Note that $m_{t,u,t} = 1$ implies that a train unit is never reallocated: it is moved to its departure track at time t, which means the arrival track and departure track are the same. The $m_{t,u,z}$-variables are used to identify and set the timing of the reallocation of the matchings. Finally, we introduce variables $\gamma_{t,u}$ and $\gamma_{t,u,f}$, that are concerned with whether matchings are reallocated or not. We define $\gamma_{t,u}$ as a binary variable which is only equal to 1 if train unit t matched to train unit u, $(t,u) \in L$ reallocates during its stay in the shunting yard, and $\gamma_{t,u,f}$ as a binary variable equal to 1 if and only if train unit t matched to train unit u, $(t,u) \in L$ reallocates from or to track f during its stay in the shunting yard. We are more interested in $\gamma_{t,u}$, but the $\gamma_{t,u,f}$-variables aid us in setting the correct values for the $\gamma_{t,u}$-variables.

The above sets of variables account for a very large number of total variables. The formulation will contain $|T|\cdot|F|$ $x_{t,f}$-variables; $|F|\cdot|T|$ $b_{f,t}$-variables; $|L|\cdot|F|$ $x^1_{t,u,f}$-variables and the same number of $x^2_{t,u,f}$-variables; approximately $|L|\cdot|T|$ $m_{t,u,z}$-variables; $|L|$ $\gamma_{t,u}$-variables; and $|L|\cdot|F|$ $\gamma_{t,u,f}$ variables. Therefore, the number of columns in the constraint matrix will be in the order of magnitude of

$$2(|T|\cdot|F|)+3(|L|\cdot|F|)+|L|(1+|T|)\in\mathcal{O}(\frac{1}{4}|T|^3),$$

since generally $3|F|<|T|$. This is almost always less than the number of variables in the formulation in section 4.1, unless the shunting yard consists of only one or two tracks.

The formulation is now as follows:

Minimize:

$$\sum_{(t,u)\in L}\gamma_{t,u} \tag{4.20}$$

Subject to:

$$\sum_{f\in F}x_{t,f} = 1 \qquad \forall t\in T \tag{4.21}$$

$$\sum_{\substack{u:\\(t,u)\in L}}x^1_{t,u,f} = x_{t,f} \qquad \forall t\in T_+,\ f\in F \tag{4.22}$$

$$\sum_{\substack{t:\\(t,u)\in L}}x^2_{t,u,f} = x_{u,f} \qquad \forall u\in T_-,\ f\in F \tag{4.23}$$

$$\sum_{f\in F}x^1_{t,u,f} = \sum_{f\in F}x^2_{t,u,f} \qquad \forall (t,u)\in L \tag{4.24}$$

$$\gamma_{t,u,f} = \left|x^1_{t,u,f}-x^2_{t,u,f}\right| \qquad \forall (t,u)\in L,\ f\in F \tag{4.25}$$

$$\sum_{f\in F}\gamma_{t,u,f} \le 2\gamma_{t,u} \qquad \forall (t,u)\in L \tag{4.26}$$

$$\gamma_{t,u} \le \frac{1}{2}\sum_{f\in F}\gamma_{t,u,f} \qquad \forall (t,u)\in L \tag{4.27}$$

$$\sum_{\substack{z\in T\\ t<z\le u}}m_{t,u,z} = \gamma_{t,u} \qquad \forall (t,u)\in L \tag{4.28}$$

$$\sum_{\substack{z\in T\\ t\le z\le u}}m_{t,u,z} = \sum_{f\in F}x^1_{t,u,f} \qquad \forall (t,u)\in L \tag{4.29}$$

$$\begin{aligned} m_{t,u,z}x_{u,f}+x_{t',f}-\sum_{\substack{u'\in T_-:\\(t',u')\in L,\\ u'<u}}\ \sum_{\substack{z'\in T:\\ t'\le z'\le u'}}m_{t',u',z'}\\ -\sum_{\substack{u'\in T_-:\\(t',u')\in L,\\ u'>u}}\ \sum_{\substack{z'\in T:\\ t'<z'\le u}}m_{t',u',z'} \le 1 \end{aligned} \qquad \begin{aligned} &\forall (t,u)\in L,\ z\in T: t\le z\le u,\\ &t'\in T_+: z\le t'<u, t'\ne t,\ f\in F \end{aligned} \tag{4.30}$$

$$m_{t,u,z}x_{u,f} + x_{u',f} - \sum_{\substack{t' \in T_+: \\ (t',u') \in L, \\ t' > u}} \sum_{\substack{z' \in T: \\ t' \leq z' \leq u'}} m_{t',u',z'}$$

$$- \sum_{\substack{t' \in T_+: \\ (t',u') \in L, \\ t' < u}} \sum_{\substack{z' \in T: \\ t' \leq z' \leq z \\ u < z' \leq u'}} m_{t',u',z'} \leq 1 \qquad \forall (t,u) \in L,\ z \in T : t \leq z \leq u,\ u' \in T_- : u' > u,\ f \in F \tag{4.31}$$

$$m_{t,u,z}x_{t,f} + x_{t',f} - \sum_{\substack{u' \in T_-: \\ (t',u') \in L, \\ u' < z}} \sum_{\substack{z' \in T: \\ t' \leq z' \leq u'}} m_{t',u',z'}$$

$$- \sum_{\substack{u' \in T_-: \\ (t',u') \in L, \\ u' \geq z}} \sum_{\substack{z' \in T: \\ t' < z' \leq z}} m_{t',u',z'} \leq 1 \qquad \forall (t,u) \in L,\ z \in T : t < z \leq u,\ t' \in T_+ : t < t' < z,\ f \in F \tag{4.32}$$

$$m_{t,u,z}x_{t,f} + x_{u',f} - \sum_{\substack{t' \in T_+: \\ (t',u') \in L, \\ t' \geq z}} \sum_{\substack{z' \in T: \\ t' \leq z' \leq u'}} m_{t',u',z'}$$

$$- \sum_{\substack{t' \in T_+: \\ (t',u') \in L, \\ t' < z}} \sum_{\substack{z' \in T: \\ t' \leq z' \leq t \\ z \leq z' \leq u'}} m_{t',u',z'} \leq 1 \qquad \forall (t,u) \in L,\ z \in T : t < z \leq u,\ u' \in T_- : u' \geq z,\ f \in F \tag{4.33}$$

$$b_{f,0} = 0 \qquad \forall f \in F \tag{4.34}$$

$$b_{f,t-1} + v_t k_t x_{t,f} + \sum_{\substack{(t',u') \in L: \\ t' < t \leq u'}} x_{u',f} m_{t',u',t} k_t$$

$$- \sum_{\substack{(t',u') \in L: \\ t' < t \leq u'}} x_{t',f} m_{t',u',t} k_t = b_{f,t} \qquad \forall f \in F,\ t \in T \tag{4.35}$$

$$b_{f,t} \leq l_f \qquad \forall t \in T,\ f \in F \tag{4.36}$$

$$x^1_{t,u,f} x_{u,g} k_t \leq n_{f,g} \qquad \forall (t,u) \in L,\ f,g \in F \tag{4.37}$$

$$\sum_{\substack{(t,u) \in L: \\ t < z \leq u}} m_{t,u,z} \leq w_z \qquad \forall z \in T \tag{4.38}$$

$$x_{t,f} \in \{0,1\} \qquad \forall t \in T,\ f \in F \tag{4.39}$$

$$x^1_{t,u,f} \in \{0,1\} \qquad \forall (t,u) \in L,\ f \in F \tag{4.40}$$

$$x^2_{t,u,f} \in \{0,1\} \qquad \forall (t,u) \in L,\ f \in F \tag{4.41}$$

$$m_{t,u,z} \in \{0,1\} \qquad \forall t,u,z \in T \tag{4.42}$$

$$\gamma_{t,u,f} \in \{0,1\} \qquad \forall (t,u) \in L,\ f \in F \tag{4.43}$$

$$\gamma_{t,u} \in \{0,1\} \qquad \forall (t,u) \in L \tag{4.44}$$

$$b_{f,t} \geq 0 \qquad \forall f \in F,\ t \in T \tag{4.45}$$

In this formulation, the objective function (4.20) minimizes the number of reallocations, by summing over the $\gamma_{t,u}$-variables. Constraints (4.21) assign each train unit to a track. Arriving and departing train units can be assigned to different tracks. If this is the case, the train unit will reallocate during its stay in the shunting yard. The arrival and departure tracks of the matchings are set equal to the tracks the arriving and departing units in the matchings are assigned to in constraints (4.22) and (4.23), respectively, and in combination with constraints (4.24) they make sure that train units are involved in exactly one matching. Constraints (4.25) force $\gamma_{t,u,f}$ to 1 if a matching is reallocated from or to track f during its stay in the shunting yard -that is $x^1_{t,u,f}$ is unequal to $x^2_{t,u,f}$ for a certain matching $(t,u) \in L$ and a certain track $f \in F$- and to 0 if it is

not reallocated or if the matching is not made at all -that is $x^1_{t,u,f}$ is equal to $x^2_{t,u,f}$ for a certain matching $(t,u) \in L$ and a certain track $f \in F$. Since $\gamma_{t,u,f}$ is set to 1 for two tracks f in case of reallocation in these constraints (namely for the arrival and departure track of the matching), $\sum_{f \in F} \gamma_{t,u,f}$ will be equal to 2 if the matching is reallocated and equal to 0 otherwise. To set the correct value for the $\gamma_{t,u}$-variables, we add constraints (4.26) and (4.27), which set $\gamma_{t,u}$ to 1 if the matching is reallocated, and 0 otherwise. Adding the constant $\frac{1}{2}$ to the right hand side of constraints (4.27) strengthens the LP bound. In constraints (4.28) the time of reallocation is determined in case a matching is reallocated, since the summation over $z \in T$ does not include time t, which is the z-value for which the $m_{t,u,z}$-variable would be set to 1 if a matching is made that is not reallocated. To correctly assign the value to the $m_{t,u,z}$-variable in case the matching is not reallocated, we use constraints (4.29), which contrary to constraints (4.28) do include t in the summation on the left hand side, and in combination with constraints (4.28) set $m_{t,u,z}$ to 1 for $z = t$ in case a matching is made that is not reallocated during its stay in the shunting yard. To summarize the first part of the problem, we can state that constraints (4.21)-(4.29) assign the correct values to the variables for the matchings that are made and the tracks these matchings arrive at and depart from, as well as the variables concerning the reallocation and reallocation times.

Constraints (4.30)-(4.33) prevent crossings from happening. They serve the same purpose as constraints (4.7)-(4.10), therefore we again refer to figure 4.2 in section 4.1 for a visual clarification. In constraints (4.32) and (4.33), crossings at departure from the first track are prevented, which is the track a matching is initially parked at and from which it reallocates to the final departure track. In constraints (4.30) and (4.31), crossings at departure from the second track are prevented, which is the track a matching has been reallocated to and from which it departs to leave the shunting yard for good. In these constraints, we also prevent crossings for train units that are never reallocated. These train units use only one track during their stay in the shunting yard. Therefore, in these cases, they can only be blocked at their departure from the shunting yard. As mentioned before in section 4.1, train units can be blocked at their departure from a track by train units that use the same track as arrival track or by train units that use the same track as departure track. In the former, the train units are reallocated away from the track and in the latter, train units are reallocated to this track at a certain point in time. If a train unit is not reallocated during its stay in the shunting yard, the reallocation time equals the arrival time, so this case is incorporated in the latter blocking possibility. We refer to figure 4.2 again for an overview of the above by the means of four timelines.

Summarizing, in constraints (4.30) and (4.31) we look at possible crossings when train units depart from a track to depart from the shunting yard, while in constraints (4.32) and (4.33) we look at possible crossings when train units depart from a track to reallocate. In constraints (4.30) and (4.31) we consider the case where for a matching $(t,u) \in L$, a moving time $z \in T$, and a track $f \in F$ both $m_{t,u,z}$ and $x_{u,f}$ are equal to 1, which means that the matching (t,u) departs from track f and it is moved to this track anytime between event $z-1$ and z. Then, we prevent matching placements that can possibly block the departure of train unit u. On the other hand, in constraints (4.32) and (4.33), we consider the case where for a matching $(t,u) \in L$, a moving time $z \in T$, and a track $f \in F$ both $m_{t,u,z}$ and $x_{t,f}$ are equal to 1, which means that the matching (t,u) is reallocated from track f anytime between event $z-1$ and z, where it arrived at the time of event t. Note that we have $t < z \leq u$, since for $z = t$ the matchings would not be reallocated, and these cases are incorporated in constraints (4.30) and (4.31). Again, we prevent matching placements that can possibly block the departure of train unit u. We now take a closer look at the individual constraints.

In constraints (4.30), we do not allow for a train unit $t' \in T_+$, that arrives in the shunting yard after the arrival of matching (t,u) on track f and before the departure of matching (t,u) from the shunting yard, to have track f as arrival track, unless it departs track f before matching (t,u) departs from the shunting yard and thus from track f. In case the matching t' is involved in leaves the shunting yard before matching (t,u) departs from the shunting yard, that is $u' < u$, no blocking can occur, thus we sum over all possible $m_{t',u',z'}$-variables in the first double summation. Train unit t' can be parked on track f without the risk of crossings. In case the matching t' is

involved in leaves the shunting yard after matching (t, u), that is $u' > u$, train unit t' can only be parked on track f if it reallocates before the departure of matching (t, u), and therefore we sum only over the $m_{t',u',z'}$-variables with $t' < z' \leq u$ in the second double summation. We have $z' > t'$ and not $z' \geq t'$, since $m_{t',u',z'} = 1$ for $z' = t'$ would indicate that train unit t' does not reallocate, which would result in a blocking since $u' > u$. We have $z' \leq u$, since it needs to reallocate before the departure of matching (t, u), and this is still the case if $m_{t',u',z'} = 1$ for $z' = u$ since the reallocation would then take place between events $u-1$ and u. This situation is illustrated in figure 4.2a.

For constraints (4.31), we again consider the case where for a matching $(t, u) \in L$, a moving time $z \in T$, and a track $f \in F$ both $m_{t,u,z}$ and $x_{u,f}$ are equal to 1. In this case, we do not allow for a train unit $u' \in T_-$, that departs from the shunting yard after the departure of matching (t, u) from track f, unless it arrives at track f before matching (t, u) arrives at track f or after matching (t, u) departs from the shunting yard and thus from track f. In case the matching u' is involved in arrives in the shunting yard after matching (t, u) departs from the shunting yard, that is $t' > u$, no blocking can occur, thus we sum over all possible $m_{t',u',z'}$-variables in the first double summation. Train unit u' can depart from track f without the risk of crossings. In case the matching u' is involved in arrives in the shunting yard before the departure of matching (t, u), that is $t' < u$, train unit u' can only be assigned to track f if it arrives at this track before the arrival of matching (t, u) on this track or if it arrives at this track through reallocation after the departure of matching (t, u) from this track. Therefore we sum over the $m_{t',u',z'}$-variables with $t' \leq z' \leq z$, which corresponds to the former case, and with $u < z' \leq u'$, which corresponds to the latter case, in the second double summation. We have $z' \geq t'$, which is simply a natural limit of z' since a matching cannot be reallocated before it even arrived in the shunting yard. We have $z' \leq z$, since it cannot arrive at track f after the arrival of matching (t, u) at track f without a crossing occurring. Next, we have $u < z'$ since the matching u' is involved in cannot arrive at the track through reallocation before matching (t, u) has departed the shunting yard. We have $z' \leq u'$, which is again a natural limit of z', since a train unit cannot be reallocated after its departure from the shunting yard. In case either $m_{t,u,z}$ or $x_{u,f}$ would be equal to 0, constraints (4.30) and (4.31) impose no restrictions on other train units parked on track f. This situation is illustrated in figure 4.2b.

Next, in constraints (4.32), we do not allow for a train unit $t' \in T_+$, that arrives in the shunting yard after the arrival of matching (t, u) on track f at time t and before the reallocation of matching (t, u) from track f to its departure track, to have track f as arrival track, unless it departs track f before matching (t, u) departs from track f when reallocating between event $z - 1$ and z. In case the matching t' is involved in leaves the shunting yard before matching (t, u) departs from track f, that is $u' < z$, no blocking can occur, thus we sum over all possible $m_{t',u',z'}$-variables in the first double summation. Train unit t' can be parked on track f without the risk of crossings. In case the matching t' is involved in leaves the shunting yard after matching (t, u) leaves track f, that is $u' \geq z$, train unit t' can only be parked on track f if it reallocates before matching (t, u) reallocates, and therefore we sum only over the $m_{t',u',z'}$-variables with $t' < z' \leq z$ in the second double summation. This situation is illustrated in figure 4.2c.

Subsequently, in constraints (4.33) we do not allow for a train unit $u' \in T_-$ that departs from the shunting yard after the reallocation of matching (t, u) from track f at time z, to have track f as departure track, unless it arrives at track f before matching (t, u) arrives at track f, or after matching (t, u) departs from track f. In case the matching u' is involved in arrives in the shunting yard after matching (t, u) departs from track f, that is $t' \geq z$, no blocking can occur, thus we sum over all possible $m_{t',u',z'}$-variables in the first double summation in the constraint and train unit u' can be assigned to track f without the risk of crossings. In case the matching u' is involved in arrives at track f before matching (t, u) leaves track f, that is $t' < z$, train unit u' can only be assigned to track f if it arrives on track f before matching (t, u) arrives there or if it arrives at this track through reallocation after the departure of matching (t, u) from this track. Therefore we sum over the $m_{t',u',z'}$-variables with $t' \leq z' \leq t$, which corresponds to the former case, and with $z \leq z' \leq u'$, which corresponds to the latter case, in the second double summation in the constraint. We have $z' \geq t'$, where t' is simply the natural limit of z' on the down side, and we

have $z' \leq t$, since the matching u' is involved in should arrive before matching (t, u). Next, we have $z \leq z'$ since the matching u' is involved in cannot arrive at the track through reallocation before matching (t, u) has departed the track. We have $z' \leq u'$, which is again a natural limit of z', since a train unit cannot be reallocated after its departure from the shunting yard. In case either $m_{t,u,z}$ or $x_{t,f}$ would be equal to 0, constraints (4.32) and (4.33) impose no restrictions on other train units parked on track f. This situation is illustrated in figure 4.2d.

The occupied track length is updated and restricted in constraints (4.34)-(4.36). Constraints (4.34) simply set the initial values of the occupied track lengths to 0. Constraints (4.35) have a more complicated nature. The occupied track length at time $t \in T$ is updated by adding or subtracting the length of the train unit that respectively arrives or departs at time t if it is assigned to the track. Then, the cumulative length of all matchings that are reallocated to the track between time $t-1$ and t is added, and the cumulative length of all matchings that are reallocated away from the track between time $t-1$ and t are subtracted. We only sum over $m_{t',u',t}$ with (t', u') such that $t' < t < u'$, because for $t' = t$ or $u' = t$ the occupied track length is already updated in the second term of the left hand side. Next, the occupied track length at all times is limited to the actual track length in constraints (4.36).

In constraints (4.37), the length of a matching moving between two tracks is limited to the maximum drive back length, while constraints (4.38) limit the number of reallocations that can take place between two events. Finally, constraints (4.39)-(4.45) set the domains of all used variables.

The number of constraints in the formulation discussed in this section is in the order of magnitude of

$$2|T| + |F| + 5|L| + 3(|T| \cdot |F|) + 6(|L| \cdot |T| \cdot |F|) + (1 + |F|)(|L| \cdot |F|) \in \mathcal{O}(\frac{3}{2}|T|^3 \cdot |F|).$$

However, the number of rows in this matrix will likely be increased since we need to linearize the non-linear constraints.

4.2.1 Resolving non-linearity issues

An issue we briefly mentioned in the introductory paragraphs of this section is the non-linearity of this model: constraints (4.30)-(4.33), (4.35), and (4.37) contain multiplications of variables, which makes these constraints non-linear which can be an issue when applying most common solution methods. Furthermore, constraints (4.25) contains an absolute value, which also complicates exact solution methods.

The issue of non-linearity can usually be resolved rather easily. If we are dealing with a multiplication of binary variables x_1 and x_2, we define $q = x_1 x_2$, which is also a binary variable. Subsequently, we define the following constraints using q, x_1, and x_2:

$$\begin{aligned} q &\leq x_1 \\ q &\leq x_2 \\ q &\geq x_1 + x_2 - 1 \\ q, x_1, x_2 &\in \{0, 1\} \end{aligned}$$

The above constraints will always assign the correct value to q, since it will be equal to 0 if either x_1 or x_2 equals zero, and equal to 1 if both x_1 and x_2 equal 1. This will result in approximately $|L| \cdot |T| \cdot |F| + |L| \cdot |F|^2$ extra variables and three times this number of extra constraints.

The absolute value issue that arises in constraints (4.25) is resolved by replacing this constraint by

the following set of constraints:

$$\gamma_{t,u,f} \geq x^1_{t,u,f} - x^2_{t,u,f} \qquad \forall (t,u) \in L,\ f \in F \tag{4.46}$$
$$\gamma_{t,u,f} \geq x^2_{t,u,f} - x^1_{t,u,f} \qquad \forall (t,u) \in L,\ f \in F \tag{4.47}$$
$$\gamma_{t,u,f} \leq x^1_{t,u,f} + x^2_{t,u,f} \qquad \forall (t,u) \in L,\ f \in F \tag{4.48}$$
$$\gamma_{t,u,f} \leq 2 - x^1_{t,u,f} - x^2_{t,u,f} \qquad \forall (t,u) \in L,\ f \in F \tag{4.49}$$

If a matching $(t,u) \in L$ reallocates during its stay in the shunting yard and $x^1_{t,u,f}$ is thus unequal to $x^2_{t,u,f}$ for a certain track $f \in F$, the corresponding variable $\gamma_{t,u,f}$ is forced to 1 due to constraints (4.46) and (4.47). If matching (t,u) does not reallocate, $x^1_{t,u,f}$ and $x^2_{t,u,f}$ will either both be equal to 1 or both equal to 0 for a certain track $f \in F$ and $\gamma_{t,u,f}$ should be forced to 0. The former situation is handled by constraints (4.49) and the latter in constraints (4.48). Constraints (4.48) also ensure $\gamma_{t,u,f}$ is set to 0 if the matching is not realized. This linearization method will result in $4(|L| \cdot |F|) - |L| \cdot |F| = 3(|L| \cdot |F|)$ extra constraints.

Applying the linearization methods as discussed in this subsection will result in a constraint matrix with dimensions of the order of magnitude

$$\big[2(|T| \cdot |F|) + 3(|L| \cdot |F|) + |L|(1 + |T|) + |L| \cdot |T| \cdot |F| + |L| \cdot |F|^2\big]$$
$$\times$$
$$\big[2|T| + |F| + 5|L| + 3(|T| \cdot |F|) + 9(|L| \cdot |T| \cdot |F|) + (4 + 2|F|)(|L| \cdot |F|)\big],$$

where the number of columns is in complexity order $\mathcal{O}(\frac{1}{4}|T|^3 \cdot |F|)$ and the number of rows in complexity order $\mathcal{O}(\frac{9}{4}|T|^3 \cdot |F|)$.

4.3 Extensions

In this section, we discuss several extensions to the formulation as proposed in section 4.2. We look for example at a formulation in which free tracks are included in subsection 4.3.1 and we also look at track lay-outs with LIFO tracks on both sides of the diagonal connecting track in subsection 4.3.2. Next, we look into other more complicated track layouts and their effect on drive back distances in subsection 4.3.3. Besides this, in subsection 4.3.4, we discuss how we can validate the formulation to also be applicable to multiple train units per train service, which will result in couple and decouple actions. Although the subsections on the possible extensions of our formulation are all written to be applied to the global formulation as proposed in section 4.2, they can be combined as well when some adaptions are made.

4.3.1 Free tracks extension

The formulation in section 4.2 only considers shunting yards consisting of only LIFO tracks on one side of the diagonal track. However, we sometimes encounter shunting yards with free tracks. Figure 4.3 gives an example of such a shunting yard layout. One can easily see that a layout like the one in figure 4.3 provides us with more routing opportunities. This requires us to define some extra parameters and variables.

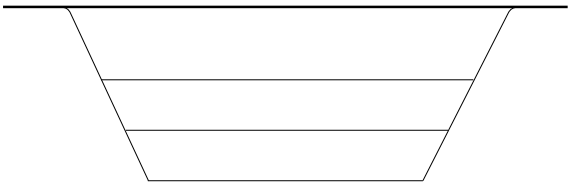


Figure 4.3: Example track layout only free tracks

		$\delta_{t,u,f} < \delta_{t',u',f}$		$\delta_{t',u',f} < \delta_{t,u,f}$	
		$\psi^1_{t,u}, \phi_u = 0$	$\psi^1_{t,u}, \phi_u = 1$	$\psi^1_{t',u'}, \phi_{u'} = 0$	$\psi^1_{t',u'}, \phi_{u'} = 1$
$\alpha_{t,u,f} < \alpha_{t',u',f}$	$\psi^2_{t',u'}, \phi_{t'} = 0$	Crossing	No crossing	No crossing	Crossing
	$\psi^2_{t',u'}, \phi_{t'} = 1$	No crossing	Crossing	Crossing	No crossing
$\alpha_{t',u',f} < \alpha_{t,u,f}$	$\psi^2_{t,u}, \phi_t = 0$	No crossing	Crossing	Crossing	No crossing
	$\psi^2_{t,u}, \phi_t = 1$	Crossing	No crossing	No crossing	Crossing

Table 4.2: Crossing overview free tracks

We have parameter ϕ_t, which is equal to 0 if train unit $t \in T$ arrives or departs via the A-side of the shunting yard (and thus the A-side of its arrival or departure track) and equal to 1 if train unit $t \in T$ arrives or departs via the B-side. Naturally, in case $t \in T_+$, ϕ_t refers to the arrival side of the train unit at the arrival track it is assigned to, and in case $t \in T_-$, ϕ_t refers to the departure side of the train unit from the departure track it is assigned to, so to the track f such that $x_{t,f} = 1$. In case the arrival or departure side of a train unit is not predetermined in the schedule, ϕ_t becomes a variable rather than a parameter, meaning it will be chosen most conveniently with respect to our objective value. In figure 4.3, this would be the case if it were allowed for a train unit arriving from the A-side to pass the first shunting yard entrance it encounters, subsequently pass the second entrance it encounters, reverse driving directions, and enter via the second entrance on the B-side, and vice versa. However, this is generally not allowed and we will not be discussed further in this extension.

Furthermore, we have variables $\psi^1_{t,u}$ and $\psi^2_{t,u}$. The former is equal to 0 if train unit $t \in T_+$ matched to train unit $u \in T_-$ departs via the A-side during reallocation, and equal to 1 if train unit $t \in T_+$ matched to train unit $u \in T_-$ departs via the B-side during reallocation, with of course $(t,u) \in L$. $\psi^2_{t,u}$ is defined similarly, but is concerned with the arrival side during reallocation and is equal to 0 for arrival at the A-side during reallocation and equal to 1 for arrival at the B-side during reallocation if train unit $t \in T_+$ is matched to train unit $u \in T_-$, $(t,u) \in L$. Naturally, we have $\psi^1_{t,u} = \psi^2_{t,u}$ for all matchings (t,u), meaning that when reallocating, the departure side from the arrival track must be equal to the arrival side on the departure track. This is generally true, since it is usually impossible to switch shunting yard sides when reallocating, as we cannot use the main railway network. However, in case there is parking track unoccupied at the time of reallocation and we can feasible route via this track when reallocating, $\psi^1_{t,u}$ and $\psi^2_{t,u}$ could differ. However, this is outside the scope of this research. If no reallocation occurs, we set $\psi^2_{t,u} = \phi_t$ and $\psi^1_{t,u} = \phi_u$ if train units t and u are matched to each other.

To get an overview of the constraints the introduction of free tracks imposes, we took a look at all possible situations that can occur when two train units spend time on the same track during their stay in the shunting yard. The overview is given in table 4.2. We again use $\alpha_{t,u,f}$ and $\delta_{t,u,f}$ as defined in section 4.1 for clarification purposes. As mentioned before in section 4.1, the situations illustrated in table 4.2 only hold for matchings $(t,u),(t',u') \in L$ that are **a)** made and **b)** occupy a track f simultaneously for any period of time in the planning period. Furthermore, if we consider the first track a matching (t,u) occupies during its stay in the shunting yard, we are dealing with variables ϕ_t and $\psi^1_{t,u}$, and if it is the second track, these are the variables ϕ_u and $\psi^2_{t,u}$.

When focusing on the cases such that $\delta_{t,u,f} < \delta_{t',u',f}$, we see that in order to avoid crossings, the arrival side of matching (t',u') cannot be equal to the departure side of matching (t,u) if $\delta_{t,u,f} < \delta_{t',u',f}$ and $\alpha_{t,u,f} < \alpha_{t',u',f}$. If $\delta_{t,u,f} < \delta_{t',u',f}$ and $\alpha_{t',u',f} < \alpha_{t,u,f}$, the arrival and departure sides of matching (t,u) must be equal since we intend to prevent crossings. We only need to formulate the constraints for one of the cases, since the other ones will be automatically included for the opposite choices for (t,u) and (t',u'). We focus on the cases such that $\delta_{t,u,f} < \delta_{t',u',f}$. Attempts to formulate the constraints in a similar manner as in constraints (4.30)-(4.33) result in very elaborate and complicated constraints. We again have four situations in which crossings can occur. A train unit can be blocked at departure from its initial or departure track, by a train unit for which the track is either its initial or its departure track. However, crossings do not only occur when $\alpha_{t,u,f} < \alpha_{t',u',f}$, but also when $\alpha_{t,u,f} > \alpha_{t',u',f}$, since arrival and departure sides

co-determine the situations in which crossings occur. These situations are illustrated in figure 4.4. Again, they only hold if the train units spend time together on the same track, so not in case $\alpha_{t',u',f} > \delta_{t,u,f}$. We will now take the situation where a train unit is blocked at final departure from the shunting yard at its final departure track as in figure 4.4c and figure 4.4d and formulate a set of constraints to replace constraints (4.31).

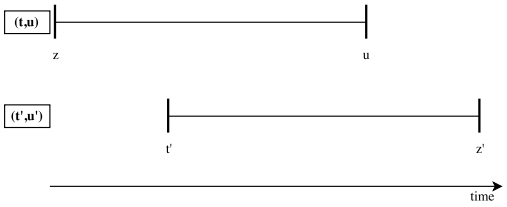
$$m_{t,u,z}x_{u,f} + x_{u',f} - \sum_{\substack{t' \in T_+: \\ (t',u') \in L}} \Big(\sum_{\substack{z' \in T: \\ u < z' \leq u'}} m_{t',u',z'} + \sum_{\substack{z' \in T: \\ z \leq z' \leq u}} m_{t',u',z'}(\left|\phi_u - \psi^2_{t',u'}\right|) + \sum_{\substack{z' \in T: \\ z' \leq z}} m_{t',u',z'}(1 - \left|\phi_u - \psi^2_{t,u}\right|)\Big) \leq 1$$
$$\forall (t,u) \in L,\ z \in T : t \leq z \leq u,\ u' \in T_- : u' > u,\ f \in F \qquad (4.50)$$

In constraints (4.50), we use the idea that we prohibit a combination of two train units on a track, unless certain conditions hold. In case matching (t,u) uses track f as final departure track and the same goes for train unit u' with $u' > u$, both $m_{t,u,z}x_{u,f}$ and $x_{u',f}$ are equal to 1 for a certain value of z. This makes their sum equal to 2, and if we would not subtract any other terms, this combination would not be allowed since $2 > 1$. However, we subtract terms that refer to the cases in which this combination of (t,u) and the matching u' is involved in is allowed, which are the following in order of appearance:

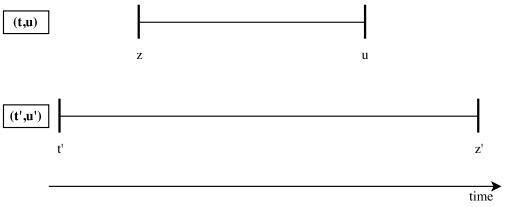
1. They do not spend time together on the track ($\alpha_{t',u',f} > \delta_{t,u,f}$);
2. The matching u' is involved in arrives at the track after (t,u) and before (t,u) departs, and the arrival side of the matching u' is involved in is not equal to the departure side of (t,u) ($\alpha_{t,u,f} < \alpha_{t',u',f} < \delta_{t,u,f} \wedge \phi_u \neq \psi^2_{t',u'}$);
3. The matching u' is involved in arrives at the track before (t,u), and the arrival side of (t,u) is equal to the departure side of (t,u) ($\alpha_{t,u,f} > \alpha_{t',u',f} \wedge \phi_u = \psi^2_{t,u}$).

The terms that refer to cases that do not result in crossings consist of a summation over $m_{t',u',z'}$ and possibly an extra element. The choice of bounds of the summations over $m_{t,u,z}$ determines whether $\alpha_{t',u',f} > \delta_{t,u,f}$, $\alpha_{t,u,f} < \alpha_{t',u',f} < \delta_{t,u,f}$, or $\alpha_{t,u,f} > \alpha_{t',u',f}$ and thus which case we are dealing with. The extra element is concerned with the arrival and/or departure sides of the train units. In the first case, the combination of (t,u) and the matching u' is involved can never be involved in a crossing, so we do not need any extra elements in the term. In the second case, we need the arrival side of the matching u' is involved to be different than the departure side of (t,u), which means we multiply by the absolute value of the difference between the two ($\psi^2_{t',u'}$ and ϕ_u). If they are equal to each other, this will be equal to 0, which means the term cancels out, the summation over $m_{t',u',z'}$ is not subtracted, and the two train units are not allowed on the same track. If they are unequal, this will be equal to 1, and the summation over $m_{t',u',z'}$ is subtracted meaning the two train units are allowed on the same track. In the third case, we need the arrival side of (t,u) to be equal to the departure side of (t,u), which means we multiply by 1 minus the absolute value of the difference the two (ϕ_u and $\psi^2_{t,u}$). If they are unequal to each other, this will be equal to $1 - 1 = 0$, which means the term cancels out, the summation over $m_{t',u',z'}$ is not subtracted, and the two train units are not allowed on the same track. If they are unequal, this will be equal to $1 - 0 = 1$, and the summation over $m_{t',u',z'}$ is subtracted meaning the two train units are allowed on the same track.

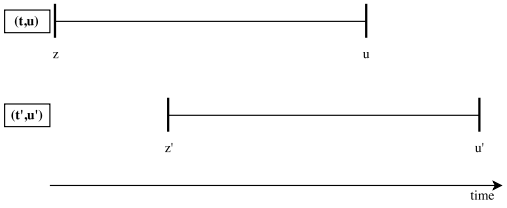
Constraints (4.50) are complicated due to the strict and precise boundaries on the summations and their non-linearity. The non-linearity is caused by the absolute value and the product of multiple variables. We can resolve this with the methods discussed in section 4.2.1. First, we need to add extra binary variables equal to the the absolute value of the difference of two arrival and/or departure side variables and add the constraints similar to the ones discussed for constraints (4.25) to assign the correct values to these variables. Subsequently, we define extra variables for the product of these newly created binary variables and $m_{t',u',z'}$ and add the appropriate constraints. In section 4.2.1 a method for linearizing the product of two binary variables is given that can



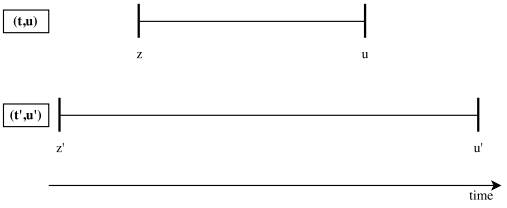
(a) Departure track (t, u) equals arrival track (t', u'), crossing if $\phi_u = \phi_{t'}$



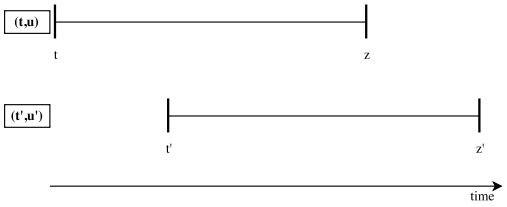
(b) Departure track (t, u) equals arrival track (t', u'), crossing if $\psi^2_{t,u} \neq \phi_u$



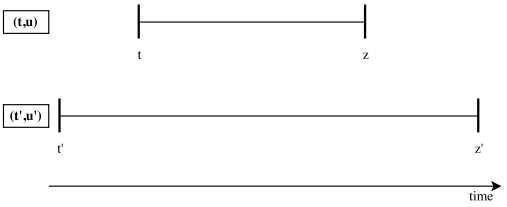
(c) Departure track (t, u) equals departure track (t', u'), crossing if $\phi_u = \psi^2_{t',u'}$



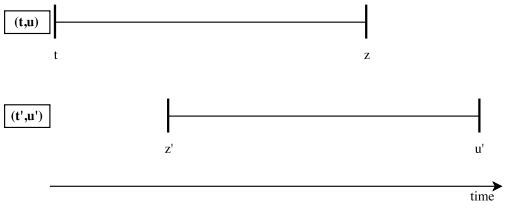
(d) Departure track (t, u) equals departure track (t', u'), crossing if $\phi_u \neq \psi^2_{t,u}$



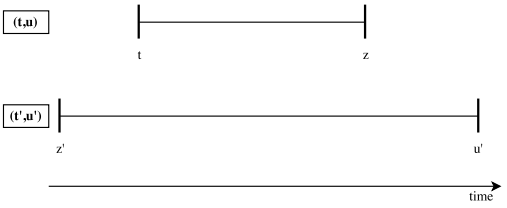
(e) Arrival track (t, u) equals arrival track (t', u'), crossing if $\psi^1_{t,u} = \phi_{t'}$



(f) Arrival track (t, u) equals arrival track (t', u'), crossing if $\phi_t \neq \psi^1_{t,u}$



(g) Arrival track (t, u) equals departure track (t', u'), crossing if $\psi^1_{t,u} = \psi^2_{t',u'}$



(h) Arrival track (t, u) equals departure track (t', u'), crossing if $\phi_t \neq \psi^1_{t,u}$

Figure 4.4: Crossing possibilities free tracks, with on the left situations such that $\alpha_{t,u,f} < \alpha_{t',u',f}$ and on the right $\alpha_{t,u,f} > \alpha_{t',u',f}$; on both sides $\delta_{t,u,f} < \delta_{t',u',f}$

be applied here. However, this results in many extra constraints and does not result in easy, intuitive constraints. Therefore, we propose a different formulation of the constraints to replace constraints (4.30)-(4.33) in section 4.2 that is similar to the formulation of the crossing constraints in section 4.1 below.

$$m_{t,u,z}x_{u,f} + m_{t',u',z'}x_{t',f} \leq 1 + |\phi_u - \phi_{t'}| \qquad \forall (t,u),(t',u') \in L,\ z,z' \in T : z \leq t' < u < z',\ t \leq z \leq u, t' < z' \leq u' \tag{4.51}$$

$$m_{t,u,z}x_{u,f} + m_{t',u',z'}x_{t',f} \leq 2 - \left|\psi^2_{t,u} - \phi_u\right| \qquad \forall (t,u),(t',u') \in L,\ z,z' \in T : t' < z \leq u < z',\ t \leq z \leq u, t' < z' \leq u' \tag{4.52}$$

$$m_{t,u,z}x_{u,f} + m_{t',u',z'}x_{u',f} \leq 1 + \left|\phi_u - \psi^2_{t',u'}\right| \qquad \forall (t,u),(t',u') \in L,\ z,z' \in T : z < z' \leq u < u',\ t \leq z \leq u, t' \leq z' \leq u' \tag{4.53}$$

$$m_{t,u,z}x_{u,f} + m_{t',u',z'}x_{u',f} \leq 2 - \left|\psi^2_{t,u} - \phi_u\right| \qquad \forall (t,u),(t',u') \in L,\ z,z' \in T : z' < z \leq u < u',\ t \leq z \leq u, t' \leq z' \leq u' \tag{4.54}$$

$$m_{t,u,z}x_{t,f} + m_{t',u',z'}x_{t',f} \leq 1 + \left|\psi^1_{t,u} - \phi_{t'}\right| \qquad \forall (t,u),(t',u') \in L,\ z,z' \in T : t < t' < z < z',\ t < z \leq u, t' < z' \leq u' \tag{4.55}$$

$$m_{t,u,z}x_{t,f} + m_{t',u',z'}x_{t',f} \leq 2 - \left|\phi_t - \psi^1_{t,u}\right| \qquad \forall (t,u),(t',u') \in L,\ z,z' \in T : t' < t < z < z',\ t < z \leq u, t' < z' \leq u' \tag{4.56}$$

$$m_{t,u,z}x_{t,f} + m_{t',u',z'}x_{u',f} \leq 1 + \left|\psi^1_{t,u} - \psi^2_{t',u'}\right| \qquad \forall (t,u),(t',u') \in L,\ z,z' \in T : t < z' < z \leq u',\ t < z \leq u, t' \leq z' \leq u' \tag{4.57}$$

$$m_{t,u,z}x_{t,f} + m_{t',u',z'}x_{u',f} \leq 2 - \left|\phi_t - \psi^1_{t,u}\right| \qquad \forall (t,u),(t',u') \in L,\ z,z' \in T : z' \leq t < z \leq u',\ t < z \leq u, t' \leq z' \leq u' \tag{4.58}$$

$$\psi^1_{t,u} = \psi^2_{t,u} \qquad \forall (t,u) \in L \tag{4.59}$$

Constraints (4.51)-(4.58) prohibit the crossing situations in figure 4.4a-4.4h. They forbid to place both matching (t,u) with reallocation time z and (t',u') with reallocation time z' on the same track if they spend time together on this track (indicated by the domain of the variables for which the constraints hold), unless the arrival and departure sides make sure no crossing occurs. In case we stated in figure 4.4 that crossings occur if the departure side of (t,u) equals the arrival side of (t',u'), the right hand side of the constraint equals 1 plus the absolute value of the difference between the sides. If the sides are the equal to each other, the right hand side is equal to 1 and only one of the two terms $m_{t,u,z}x_{t,f}$ (or $m_{t,u,z}x_{u,f}$) and $m_{t',u',z'}x_{t',f}$ (or $m_{t',u',z'}x_{t',f}$) can be equal to 1. Otherwise, the constraints are not restricting, since the right hand side is equal to 2, thus they can both be equal to 1 and can be placed on the same track. In case we stated in figure 4.4 that crossings occur if the arrival and departure side of (t,u) differ, the right hand side of the constraint equals 2 minus the absolute value of the difference between the sides. If the sides differ, the right hand side is equal to 1 and only one of the two terms $m_{t,u,z}x_{t,f}$ (or $m_{t,u,z}x_{u,f}$) and $m_{t',u',z'}x_{t',f}$ (or $m_{t',u',z'}x_{u',f}$) can be equal to 1. Otherwise, the right hand side is equal to 2 and the constraints are again not restricting. Constraints (4.51)-(4.58) still contain non-linear elements, but these non-linear elements are easily linearized cases as discussed in section 4.2.1 and do not contain a non-linear element which is the multiplication of other non-linear elements as in constraints (4.50). Finally, constraints (4.59) state that the departure side from the arrival track must be equal to the arrival side on the departure track when reallocating, as discussed before in this section.

If we are dealing with both LIFO and free tracks, we will need some special constraints for the LIFO tracks. We now define $F_A \subset F$ the set of LIFO tracks opened at the A-side, and $F_B \subset F$ the set of LIFO tracks opened at the B side. We can now formulate the following constraints in order to ensure the correct usage of LIFO tracks.

$$x_{t,f} + \phi_t \leq 1 \qquad \forall t \in T,\ f \in F_A \tag{4.60}$$

$$x_{t,f} - \phi_t \leq 0 \qquad \forall t \in T,\ f \in F_B \tag{4.61}$$

$$x^1_{t,u,f} + \psi^1_{t,u} \leq 1 \qquad \forall (t,u) \in L,\ f \in F_A \tag{4.62}$$

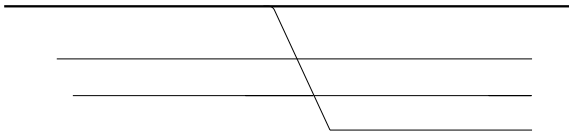


Figure 4.5: Example track layout LIFO tracks on both sides of the diagonal track

$$x^1_{t,u,f} - \psi^1_{t,u} \leq 0 \qquad \forall (t,u) \in L,\ f \in F_B \tag{4.63}$$

$$x^2_{t,u,f} + \psi^2_{t,u} \leq 1 \qquad \forall (t,u) \in L,\ f \in F_A \tag{4.64}$$

$$x^2_{t,u,f} - \psi^2_{t,u} \leq 0 \qquad \forall (t,u) \in L,\ f \in F_B \tag{4.65}$$

In constraints (4.60) and (4.61) the arrival or departure side for a train unit at initial arrival in or final departure from the shunting yard is forced to the appropriate side. It is forced to the A-side for tracks that are open at the A-side in constraints (4.60): if $x_{t,f} = 1$ for a train unit $t \in T$ and a track $f \in F_A$, ϕ_t is forced to 0 to meet the restriction. On the other hand, it is forced to the B-side for B-side opened tracks in constraints (4.61): if $x_{t,f} = 1$ for a train unit $t \in T$ and a track $f \in F_B$, ϕ_t is forced to 1 to satisfy the constraint. Constraints (4.62) and (4.63) work in a similar manner for the side train units use for reallocation from LIFO tracks, and constraints (4.64) and (4.65) assign appropriate arrival sides for reallocations of matchings to LIFO tracks.

Lastly, the positioning of train units on a track is of importance too when parking on a free track: if a train unit arriving via the A-side of a track is parked close to the B-side of this track, there might be enough track length available for a new train unit arriving from the B-side, but the first train unit should be moved more to the A-side of the track for the second train unit to use this available space. However, that is outside the scope of this research.

4.3.2 LIFO tracks on both sides of diagonal track extension

Where we consider only track layouts with LIFO tracks on one side of the diagonal track as in figure 4.1, shunting yards with LIFO tracks on both side of the diagonal track exist as well, as in figure 4.5. If we want to park train units on the left hand side of the diagonal track in figure 4.5, we need to use the diagonal track to drive past the intended parking track and then reverse the driving direction to enter the track. For departure from the track the opposite route is followed. This means there exist restrictions on the maximum length a train unit can have when assigning it to the tracks on the left side of the diagonal track: they cannot be longer than the distance between the track and the end of the diagonal track. We denote this distance by parameter $q_f \forall f \in F$. For tracks on the right hand side where no restrictions on train unit length apply, we set q_f to a large value $M \gg \max_{t \in T} k_t$ so the corresponding constraints will not be restricting. Now, we introduce the following set of constraints:

$$k_t x_{t,f} \leq q_f \qquad \forall f \in F \tag{4.66}$$

By adding constraints (4.66) to the formulation in section 4.2, we also account for shunting yard layouts similar to the one displayed in figure 4.5.

4.3.3 Flexible maximum train unit reallocation length

Until now, we have assumed that we have a fixed maximum train unit length that can move between two certain tracks, which is logical if we assume a track layout similar to the ones in figure 4.1 and figure 4.3. However, this might not always be the case, this length may vary. This is best explained by means of an example. Consider the track layout in figure 4.6. If we want to reallocate a train unit from the red to the green track in this figure (provided there is enough room available on the green track), we can drive back on the black diagonal track connected to the red and green track, and subsequently enter the green track. However, it is likely the train unit will be too long to drive back on the black diagonal track up to the green track without using the main railway network, which is not allowed. In this case, it might be possible to reallocate to the blue track,

which can be entered directly from the red track, using the part of the diagonal track between the red and blue track and subsequently crossing the diagonal track to reach the green track. This is only possible if there is enough space available on the blue track. The available space on the blue track may vary however, due to the cumulative length of train units already parked on this track. Another possibility is to reallocate using the diagonal track and the pink track. In this case, enough space must be available on the pink track. However, we do not need as much space as on the blue track, since only part of the train unit needs to access the pink track: as soon as the train unit has passed the intersection between the diagonal and green track, the driving direction can be reversed and the train unit can access the green track. In order to incorporate this, the maximum length of a train unit to feasibly move from one track to another, $n_{f,g}$, $f, g \in F$, as defined in chapter 3 will be a variable rather than a parameter, which will need to be updated with each event. This means it will also need the extra index $z \in T$, the time of the event. We will use an asterisk in its notation as a variable ($n^*_{f,g,z}$) to avoid confusion. Now, $n^*_{f,g,z}$ is equal to the maximum length for a train unit reallocating between track f and g between event z and $z-1$.

We need to define the set O, which contains the pairs of tracks $f, g \in F$ with a fixed maximum train unit length to reallocate from track f to g. This means that $n^*_{f,g,z}$ should be equal to $n_{f,g}$ for all $z \in T$ if $(f, g) \in O$. The set O also contains the pairs of tracks where the allowed train unit length to reallocate between them is unrestricted, as is the case when reallocating between the blue and red track in figure 4.6. Next, we define the set J, which contains all routes between all possible pairs of tracks f and g such that $(f, g) \notin O$. The set $J_{f,g} \subset J$ contains all routes between track f and g. We also define the set $F_j \subset F$, which contains for a route $j \in J$ the parking tracks $h \in F$ that are used in this route. F_j often consists of only one track, but when for example reallocating from the yellow to the blue track in figure 4.6, a possible route is to exit the yellow track and drive straight onto the pink track, use the diagonal track to reach the green track, and from there on access the blue track. The parameter $d_{j,h}$ equals the length of the part of the train unit that does not need to enter track h when reallocating via route j, as would be the case for the pink track when reallocating from the red to the green track via the pink track in figure 4.6 as explained in the example. Parameter e_j equals $n_{f,g}$ in case route $j \in J_{f,g}$ does not use any parking tracks when reallocating, but only uses the diagonal track as in the original formulation. Otherwise, it is equal to 0. Now, to assign the correct values to $n^*_{f,g,z}$, we need the following constraints:

$$n^*_{f,g,z} = n_{f,g} \qquad \forall f, g \in F : (f, g) \in O,\ z \in T \tag{4.67}$$

$$n^*_{f,g,z} = \max_{j \in J_{f,g}} \{\min_{h \in F_j} \{l_h - b_{h,z-1} + d_{j,h}\}, e_j\} \qquad \forall f, g \in F : (f, g) \notin O,\ z \in T \tag{4.68}$$

$$n^*_{f,g,z} \in \mathbb{R} \qquad \forall f, g \in F,\ z \in T \tag{4.69}$$

$$n^*_{f,g,z} \geq 0 \qquad \forall f, g \in F,\ z \in T \tag{4.70}$$

Here, constraints (4.67) assign a fixed value to $n^*_{f,g,z}$ for the pairs of tracks with a fixed maximum train unit length to reallocate from one to another. Constraints (4.68) update the value of $n^*_{f,g,z}$: they set the value of $n^*_{f,g,z}$ to the largest of the maximum allowed train unit lengths on all possible routes between track f and track g. The maximum allowed train unit length of a route is equal to the smallest available length on the parking tracks that are part of the route if the train unit fully enters the parking track on the route. If the train unit does not fully enter the track, we add $d_{j,h}$. We formulate this as $l_h - b_{h,z-1} + d_{j,h}$. Note that we use $z - 1$ as an index for the occupied track length $b_{h,z-1}$, since $b_{h,z-1}$ is equal to the occupied track length of track h right after event $z - 1$. Since $n^*_{f,g,z}$ is the maximum train unit reallocation length between event $z - 1$ and event z, this works out. If no parking tracks are used on route j, the maximum allowed train unit reallocation

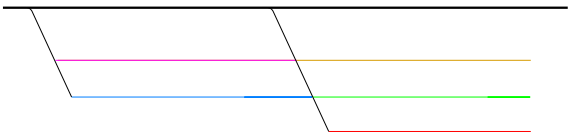


Figure 4.6: Example track layout varying drive back distance

length on this route is equal to e_j as discussed before. Finally, constraints (4.69) and (4.70) define the domain for $n^*_{f,g,z}$.

As can be seen fairly easy, constraints (4.68) are non-linear. We start their linearization by replacing the expression within the maximum value function, that is $\{\min_{h \in F_j}\{l_h - b_{h,z-1} + d_{j,h}\}, e_j\}$, by the variable $r_{j,z}$. This variable is equal to the maximum allowed reallocation length of route $j \in J$ when reallocating between event $z-1$ and z. We assign the correct value to $r_{j,z}$ by means of the following set of constraints:

$$r_{j,z} \leq l_h - b_{h,z-1} + d_{j,h} \qquad \forall j \in J_{f,g} : F_j \neq \emptyset,\ h \in F_j,\ z \in T \tag{4.71}$$
$$r_{j,z} \leq e_j \qquad \forall j \in J_{f,g} : F_j = \emptyset,\ z \in T \tag{4.72}$$
$$r_{j,z} \in \mathbb{R} \qquad \forall j \in J,\ z \in T \tag{4.73}$$
$$r_{j,z} \geq 0 \qquad \forall j \in J,\ z \in T \tag{4.74}$$

Now, we can finish the linearization by the introduction of a binary variable $s_{j,z}$ which is equal to one if the maximum train unit reallocation length for a time z is equal to at least $r_{j,z}$, and 0 otherwise. We introduce the following set of constraints:

$$n^*_{f,g,z} \geq r_{j,z} \qquad \forall f,g \in F : (f,g) \notin O,\ j \in J_{f,g},\ z \in T \tag{4.75}$$
$$n^*_{f,g,z} \leq r_{j,z} + (1 - s_{j,z})M \qquad \forall f,g \in F : (f,g) \notin O,\ j \in J_{f,g},\ z \in T \tag{4.76}$$
$$\sum_{j \in J_{f,g}} s_{j,z} \geq 1 \qquad \forall f,g \in F : (f,g) \notin O,\ z \in T \tag{4.77}$$
$$s_{j,z} \in \mathbb{B} \qquad \forall j \in J,\ z \in T \tag{4.78}$$

Here, we have $M \gg 0$. Constraints (4.75)-(4.77) state the maximum train unit reallocation length between track f and g at time z is equal to at most the maximum of all $r_{j,z}$ such that $j \in J_{f,g}$, since $s_{j,z}$ must be equal to 1 for at least one route j such that $j \in J_{f,g}$. This does not necessarily mean that it will be equal to 1 for the route j with maximum value $r_{j,z}$: if the train unit length is smaller than the maximum allowed length, another route with a large enough maximum allowed length will do as well. However, if we need the maximum length for reallocation, $s_{j,z}$ will automatically be equal to 1 for the route j with maximum value $r_{j,z}$ for feasibility purposes. Summarizing, we can linearize the discussed extension by replacing constraints (4.68) by constraints (4.71)-(4.78).

Finally, besides only adding the constraints discussed in this section to the formulation discussed in section 4.2, we need to update constraints (4.37) to the following:

$$x^1_{t,u,f} x^2_{t,u,g} m_{t,u,z} k_t \leq n^*_{f,g,z} \qquad \forall (t,u) \in L,\ f,g \in F,\ z \in T : t < z \leq u \tag{4.79}$$

This extension makes the order in which train units are reallocated between two events very relevant, reallocations that occur earlier in the time window might result in less or rather more track capacity available for reallocation. As mentioned before we do not determine the order of reallocation in our model, which can be a problem when applying the current extension.

Constraint (4.79) is non-linear since it contains the product of three binary variables, so a linearization method similar to the one in section 4.2.1 needs to be applied. For three binary variables x_1, x_2, and x_3, we define $q_2 = x_1 x_2 x_3$, which makes q_2 a binary variable as well. The constraints belonging to this linearization can now be formulated as follows:

$$\begin{aligned} q_2 &\leq x_1 \\ q_2 &\leq x_2 \\ q_2 &\leq x_3 \\ q_2 &\geq x_1 + x_2 + x_3 - 2 \\ q_2 &\in \{0,1\} \end{aligned}$$

Just as before, the above constraints will always assign the correct value to q_2, since it will be equal to 0 if either x_1, x_2, or x_3 equals zero, and equal to 1 if all x_1, x_2, and x_3 equal 1.

4.3.4 Coupling and decoupling

Since we assume train units are decoupled right after their arrival at their initial park track and coupled right before their departure from the shunting yard, we need them to be parked in the correct order at the same departure track. Because of the specific ordering on the train units we use as explained in chapter 3, the demand regarding the order on the track is trivially satisfied. Furthermore, since we use constraints (4.38), it is not possible to reallocate between the arrival and departure of two train units belonging to the same train service. Since their arrivals or departures take place at the same time, constraints (4.38) will not allow for reallocations in between, as w_z will be equal to 0 for all event indices z of train units in the same service but the train unit that is last in the ordering. Combined with a still to be defined constraint stating that train units belonging to the same service need to be parked at the same track right after their arrival at or right before their departure from the shunting yard, this implies that no train units belonging to a different train service will be parked in between two train units belonging to the same train service right after their arrival in or departure from the shunting yard.

Now, to define the constraints needed to extend the formulation to incorporate the possibility of multiple train units per train service, we need the extra set A and the parameter a_t, $t \in T$, which are also used in [19]. Parameter a_t is defined as the train service in which train unit $t \in T$ arrives at or departs from the shunting yard and A is the set of pairs of train units t and $t+1$, $(t, t+1) \in T^2$, that arrive or depart in the same train service, that is $a_t = a_{t+1}$. We can now define the following constraint:

$$x_{t,f} = x_{t+1,f} \qquad \forall (t, t+1) \in A,\ f \in F \tag{4.80}$$

Please note that the possibility of keeping multiple train units that arrive and depart in the same composition, without actually needing to be coupled or decoupled, or the possibility to couple or decouple at a different time than right after or right before departure from the shunting yard is not incorporated. Also, this extension does not incorporate the couple and decouple times. We assume there will always be enough time to decouple train units right after the arrival of a train service at the shunting yard and to couple train units right before the departure of a train service from the shunting yard.

However, if we were to include couple and decouple times, we would need to add constraints that state that train units that are part of an arriving composition cannot be reallocated between the time of arrival and the time of arrival plus the decouple time. For train units that are part of a departing composition we would need to add constraints forbidding reallocation between the departure time minus the couple time and the departure time. We introduce the set $T^c \subset T$, which contains the train units that are part of a composition with multiple train units. Similarly as before for set T, we define sets T^c_+ and T^c_- which contain the train units that are part of an arriving or departing composition, respectively, and they are disjoint. In mathematical notation, we have:

$$T^c \subset T,\ T^c = T^c_+ \cup T^c_-,\ T^c_+ \cap T^c_- = \emptyset.$$

We now define for each train unit $t \in T^c$ the parameter χ_t, which is equal to the earliest allowed reallocation time for arriving train units and equal to the latest allowed reallocation time for departing train units. Now, we can formulate the following constraints:

$$\sum_{\substack{u \in T_-: \\ (t,u) \in L}} \sum_{\substack{z \in T: \\ t < z < \chi_t}} m_{t,u,z} = 0 \qquad \forall t \in T^c_+ \tag{4.81}$$

$$\sum_{\substack{t \in T_+: \\ (t,u) \in L}} \sum_{\substack{z \in T: \\ \chi_u < z \leq u}} m_{t,u,z} = 0 \qquad \forall u \in T^c_- \tag{4.82}$$

By summing $m_{t,u,z}$ over all possible matches $u \in T_-$ to train unit $t \in T^c_+$ and all reallocation times $z \in T$ that are forbidden and setting this expression equal to 0 in constraints (4.81), we ensure that no reallocation of train units that arrive in a composition of multiple train units takes place

until the composition has been decoupled. In the summation bound in the second summation in the constraints, we have $t < z$ and not $t \leq z$, since for $t = z$ the train unit would not reallocate at all. In constraints (4.82), we sum over all possible matches $t \in T_+$ to train unit $u \in T_-^c$ and all reallocation times $z \in T$ that are forbidden and set this expression equal to 0. This way, train units will have a reallocation time of at most χ_u.

If we would want to use a model where services with compositions of multiple train units are included as discussed in this subsection, but applied to a shunting yard layout like the one in figure 4.5 where there are LIFO tracks on both sides of the diagonal track, some extra adaptions to the existing model would need to be made. First, the entire length of the composition could not be longer than q_f when assigning track f to a train unit. To this end, we introduce the parameter c_t, which is equal to the length of the composition train unit $t \in T$ is a part of. Furthermore, we would include the following constraints instead of constraints (4.66):

$$c_t x_{t,f} \leq q_f \qquad \forall f \in F \tag{4.83}$$

Furthermore, if a composition of multiple units were to be parked on the left side in figure 4.5, the order in which they would enter the track would be different than the order in which they arrive in the shunting yard, since the moving direction is reversed. Therefore, for these tracks, the ordering of train units as discussed in section 3 will not be sufficient and we introduce an extra ordering of train units which was also introduced in [19]. In this alternative ordering, a train unit t_1 appears before train unit t_2 in the ordering (denoted by $t_1 <^* t_2$) if and only if one of the following conditions is satisfied:

1. The event time of train unit t_1 is smaller than the event time of train unit t_2;

2. Train units t_1 and t_2 are arriving train units that arrive in the same train service, and train unit t_2 is positioned more to the front of the composition than train unit t_1;

3. Train units t_1 and t_2 are departing train units that depart in the same train service, and train unit t_2 is positioned more to the front of the composition than train unit t_1.

This new ordering is particularly relevant when considering the crossing constraints and therefore we will have to adapt constraints (4.30)-(4.33). They will stay the same for tracks entered similarly as the tracks on the right side of figure 4.5, but the new ordering will be used for tracks on the left side of the diagonal track, respectively. We use sets $F_D \subset F$ and $F_N \subset F$ for the tracks on the right and left side of the diagonal track, respectively. In this notation, D stands for directly accessible, and N stands for not directly accessible. We can now formulate the set of constraints to replace constraints (4.30)-(4.33):

$$\begin{aligned} m_{t,u,z}x_{u,f} + x_{t',f} - \sum_{\substack{u' \in T_-: \\ (t',u') \in L, \\ u' < u}} \sum_{\substack{z' \in T: \\ t' \leq z' \leq u'}} m_{t',u',z'} \\ - \sum_{\substack{u' \in T_-: \\ (t',u') \in L, \\ u' > u}} \sum_{\substack{z' \in T: \\ t' < z' \leq u}} m_{t',u',z'} \leq 1 \quad & \forall (t,u) \in L,\ z \in T : t \leq z \leq u, \\ & t' \in T_+ : z \leq t' < u, t' \neq t,\ f \in F_D \end{aligned} \tag{4.84}$$

$$\begin{aligned} m_{t,u,z}x_{u,f} + x_{t',f} - \sum_{\substack{u' \in T_-: \\ (t',u') \in L, \\ u' <^* u}} \sum_{\substack{z' \in T: \\ t' \leq^* z' \leq^* u'}} m_{t',u',z'} \\ - \sum_{\substack{u' \in T_-: \\ (t',u') \in L, \\ u' >^* u}} \sum_{\substack{z' \in T: \\ t' <^* z' \leq^* u}} m_{t',u',z'} \leq 1 \quad & \forall (t,u) \in L,\ z \in T : t \leq z \leq u, \\ & t' \in T_+ : z \leq^* t' <^* u, t' \neq t,\ f \in F_N \end{aligned} \tag{4.85}$$

$$m_{t,u,z}x_{u,f} + x_{u',f} - \sum_{\substack{t'\in T_+:\\(t',u')\in L,\\ t'>u}} \sum_{\substack{z'\in T:\\ t'\le z'\le u'}} m_{t',u',z'} - \sum_{\substack{t'\in T_+:\\(t',u')\in L,\\ t'<u}} \sum_{\substack{z'\in T:\\ t'\le z'\le z\\ u<z'\le u'}} m_{t',u',z'} \le 1 \quad \forall (t,u)\in L,\ z\in T: t\le z\le u,\ u'\in T_-: u'>u,\ f\in F_D \tag{4.86}$$

$$m_{t,u,z}x_{u,f} + x_{u',f} - \sum_{\substack{t'\in T_+:\\(t',u')\in L,\\ t'>^* u}} \sum_{\substack{z'\in T:\\ t'\le^* z'\le^* u'}} m_{t',u',z'} - \sum_{\substack{t'\in T_+:\\(t',u')\in L,\\ t'<^* u}} \sum_{\substack{z'\in T:\\ t'\le^* z'\le^* z\\ u<^* z'\le^* u'}} m_{t',u',z'} \le 1 \quad \forall (t,u)\in L,\ z\in T: t\le z\le u,\ u'\in T_-: u'>^* u,\ f\in F_N \tag{4.87}$$

$$m_{t,u,z}x_{t,f} + x_{t',f} - \sum_{\substack{u'\in T_-:\\(t',u')\in L,\\ u'<z}} \sum_{\substack{z'\in T:\\ t'\le z'\le u'}} m_{t',u',z'} - \sum_{\substack{u'\in T_-:\\(t',u')\in L,\\ u'\ge z}} \sum_{\substack{z'\in T:\\ t'<z'\le z}} m_{t',u',z'} \le 1 \quad \forall (t,u)\in L,\ z\in T: t< z\le u,\ t'\in T_+: t<t'<z,\ f\in F_D \tag{4.88}$$

$$m_{t,u,z}x_{t,f} + x_{t',f} - \sum_{\substack{u'\in T_-:\\(t',u')\in L,\\ u'<^* z}} \sum_{\substack{z'\in T:\\ t'\le^* z'\le^* u'}} m_{t',u',z'} - \sum_{\substack{u'\in T_-:\\(t',u')\in L,\\ u'\ge^* z}} \sum_{\substack{z'\in T:\\ t'<^* z'\le^* z}} m_{t',u',z'} \le 1 \quad \forall (t,u)\in L,\ z\in T: t< z\le u,\ t'\in T_+: t<^* t'<^* z,\ f\in F_N \tag{4.89}$$

$$m_{t,u,z}x_{t,f} + x_{u',f} - \sum_{\substack{t'\in T_+:\\(t',u')\in L,\\ t'\ge z}} \sum_{\substack{z'\in T:\\ t'\le z'\le u'}} m_{t',u',z'} - \sum_{\substack{t'\in T_+:\\(t',u')\in L,\\ t'<z}} \sum_{\substack{z'\in T:\\ t'\le z'\le t\\ z\le z'\le u'}} m_{t',u',z'} \le 1 \quad \forall (t,u)\in L,\ z\in T: t< z\le u,\ u'\in T_-: u'\ge z,\ f\in F_D \tag{4.90}$$

$$m_{t,u,z}x_{t,f} + x_{u',f} - \sum_{\substack{t'\in T_+:\\(t',u')\in L,\\ t'\ge^* z}} \sum_{\substack{z'\in T:\\ t'\le^* z'\le^* u'}} m_{t',u',z'} - \sum_{\substack{t'\in T_+:\\(t',u')\in L,\\ t'<^* z}} \sum_{\substack{z'\in T:\\ t'\le^* z'\le^* t\\ z\le^* z'\le^* u'}} m_{t',u',z'} \le 1 \quad \forall (t,u)\in L,\ z\in T: t< z\le u,\ u'\in T_-: u'\ge^* z,\ f\in F_N \tag{4.91}$$

When combining the current extension with free tracks, one would have to add constraints stating that train units arriving or departing in the same train service also need to arrive at or depart from the same side of the track, respectively:

$$\phi_t = \phi_{t+1} \qquad \forall (t,t+1)\in A \tag{4.92}$$

Chapter 5

Solution Approach

For smaller problem instances, a CPLEX solver is able to find exact, optimal solutions using branch-and-cut methods in reasonably small computation times. Branch-and-cut is a widely used combinatorial optimization method that is used to solve integer linear programming problems by means of a combination of branch-and-bound and cutting plane methods. Shortly said, branch-and-cut methods work by sequentially solving a series of LP relaxations of the original integer programming problem. In the nodes of the branch-and-bound tree, cutting plane methods improve the LP relaxation to closer fit the original integer programming problem [22]. The LP relaxations are solved using the regular simplex method. However, the larger and more complex the problem instance, the more the computation times increase, to sometimes over 30 minutes without finding the optimal solution. This is very undesirable, which is why we developed a row generation heuristic to solve the generally large problem instances, on which the specifics can be found in section 5.1. We considered applying a column generation approach as well, but due to several considerations that can be found in section 5.2 we did not pursue this option.

5.1 Row generation heuristic

Since many variables will be equal to zero in the optimal and any other feasible solution as only one matching is realized for each train unit - meaning all other possible matchings are not made and all their associated variables will be equal to zero - crossing constraints (4.30)-(4.33) are often not restricting and thus unnecessary. These constraints are restricting if and only if we have $(t, u) \in L$, $z \in T$, and $f \in F$ such that $m_{t,u,z} = 1$ and $x_{u,f} = 1$, so in a large percentage of the cases they are superfluous and their inclusion only increases the computation time. Therefore, a row generation approach is considered. In row generation, certain constraints are excluded from the model and the remaining model is referred to as the master problem. Subsequently, the model is solved and if a solution can be found, the separation problem is addressed. The separation problem involves checking whether any of the left out constraints are violated. If this is the case, the violated constraints are added to the master problem, which is then solved again. This process is repeated until no violated constraints can be found anymore. In our model, the separation model is straightforwardly solved as it is rather easy and quick to check which of the crossing constraints in (4.30)-(4.33) is violated, and therefore we apply this technique on our model. The global steps of row generation applied to our problem are set out in algorithm 1. A sequence of models is solved in which at each iteration constraints are added for combinations of matchings and train units that are involved in crossings. If a matching (t, u) is involved in a crossing at departure from the shunting yard by matching (t', u'), constraints (4.30) are generated for the specific combination of (t, u) and t, and (4.31) are added for the specific combination of (t, u) and u. If the crossing occurs at reallocation, constraints (4.32) and (4.33) are generated for the specific matchings involved in the crossing.

Algorithm 1: Row Generation algorithm

```
Construct BasicModel without crossing constraints;
Set FeasibleSolution to false;
while FeasibleSolution is false do
    Find CurrentSolution by solving CurrentModel;
    if No solution found in CurrentSolution then
        No solution can be found overall;
        break;
    else
        Find BlockingsCurrent in CurrentSolution;
        if BlockingsCurrent = ∅ then
            Set FeasibleSolution = true;
        else
            Add crossing constraints current blockings to CurrentModel;
        end
    end
end
```

We see that in the algorithm, in each iteration we solve the model with only a subset of the constraints included and iteratively add constraints until the solution to the master problem is also feasible for the complete problem. Note that in the first iteration, the solution will never include a reallocation, since crossings are not prohibited and we minimize reallocations, unless our data contains services with compositions with multiple train units. The model is solved by CPLEX software, using branch-and-cut.

Although algorithm 1 can potentially decrease the computation times for finding a solution to our problem, the search through the solution space is not as structured as it could be and the solution structure does not use all the information it could get from the previous solution. By iteratively adding the crossing constraints, the method will seemingly randomly jump through the solution space, since it might use the current matching and add some reallocations or switch track assignments, but it might as well adapt the matching. Some preliminary tests show that we can often find a feasible solution for the first matching we find in algorithm 1 and we want to exploit that feature. By fixing the matching in the first iteration, the solution algorithm will try to prevent crossings by adding reallocations and changing track assignments in subsequent iterations. Not only do we use more information from previous iterations this way, the solution time could decrease considerably as well, since many variables will be forced to zero and excluded from the problem as we have fixed the matching. However, in case we do not find a feasible solution for the first found matching, we want to be able to explore other regions of the solution space, i.e. other matching patterns. We do this by starting with the basic model without crossing constraint, but in order to explore different regions of the solution space and to smartly make use of previous information, we need to prohibit (some elements of) the previous matching(s). We can simply prohibit the exact combination of matched train units that we previously fixed, but this might not be the most efficient approach. The algorithm could now use almost exactly the same matching pattern, with only slight differences. In many cases, we will again not be able to find a feasible solution and we might need many iterations to arrive at a matching pattern that satisfies our demands.

Let us consider the hypothetical situation where we have 20 arriving train units -i.e. 40 train units in total in our problem definition- of 4 different types. Of these 20, 9 are of type A, 2 are of type B, 4 are of type C, and 5 are of type D. We assume the arrivals and departures are not mixed in time, meaning an arriving unit can be matched to any departing unit provided it is of the correct type. In our example case, a staggering $9! \times 2! \times 4! \times 5! = 2'090'188'800$ matching patterns would exist, of which many are nearly identical. Iterating through them all has disastrous effects on the computation time. In general, if we have $|T_+|$ arriving train units of m different types, and for each of the types $1, \ldots, m$ there are $n_1, \ldots, n_m$ arriving with $\sum_{i=1}^{m} n_i = |T_+|$, there exist $n_1! \times n_2! \times \cdots \times n_m!$ different matching patterns.

So instead of randomly jumping between all corners of the solution space, focusing on the elements of the previous matching pattern(s) that most often were the cause of the infeasibility might lead to a much faster convergence to a feasible solution. Therefore we will exclude certain elements of the matching pattern, that is one or more specific matchings of arriving to departing train units that have been involved in crossings most often in previous iterations, as opposed the exact matching pattern. The matching possibilities left to be explored will be diminished considerably, and we will search our solution space more smartly. In our previous example, fixing only one matching between an arriving and departing unit of type A would lower the number of matching patterns left to be searched by a factor 9.

In our new solution method, we use a tabu search heuristic [11]. We first fix a matching and if the row generation approach applied on this subset of the solution space does not result in a feasible solution, we add the most problematic elements of this matching pattern to a tabu list. Subsequently, we return to the basic model and prohibit the matching elements from the tabu list. Of course, we do need to check whether the elements do not exclude all matching possibilities for a certain train unit. If this were the case, no feasible solution could be found in any case. If such an element is included, we remove it from the candidate tabu list.

In algorithm 2 we add matchings of arriving to departing train units that were most often involved in a crossing to the tabu list, but we will also explore the possibilities of adding the combinations that were involved in a crossing in the first iteration of the inner loop as described in algorithm 2 and/or combinations that were involved in a crossing in the last iteration. Furthermore, we can decide whether we want to prohibit combinations that were blocked by other combinations when reallocating or departing from the shunting yard, or combinations that were blocking other combinations when reallocating or departing from the shunting yard. Once a matching element is placed on the tabu list, it will never be removed. Experiments for removing elements from the tabu list after a certain number of iterations is outside the scope of this research. The general steps for this tabu search-row generation approach can be found in the overview of algorithm 2.

In algorithm 2, the tabu list is denoted by `ProhibitedMatchings`. We fix and prohibit a matching element $(t, u) \in L$ by adding a constraint that sets $\sum_{f \in F} x^1_{t,u,f}$ equal to 1 or 0, respectively. Note that the algorithm described in algorithm 2 stops after finding the first feasible solution and thus will not necessarily find the optimal solution. Also, note that the approach described in algorithm 2 is heuristic and not exact, since we use a tabu search.

The underlying principles used in the solution method we described here were taken from Bender's decomposition method as described in [1]. After solving the master problem without crossing constraints generated, we fix some variables to their current value (the matching) and solve the separation problem. However, after solving the separation problem, we do not immediately return to the master problem and 'unfix' the fixed variables, but we continue solving the separation problem until we find a feasible solution, or until there are no constraints left to be added. In the latter case, we return to the original basic master problem to which we do not add the violated constraints found in the matching problem, but we alter it by prohibiting some elements of the previously fixed matching pattern(s) by fixing the values of their associated variables equal to zero. Then, the procedure is repeated.

5.2 Consideration of column generation

The overarching idea behind column generation is that in the optimal solution, but actually in any feasible solution to our problem since we have almost solely binary variables, many variables will have an associated value of zero. Therefore, we choose to exclude a number of variables and their associated columns in the objective function and constraint matrix from the problem, to make it more computationally tractable. The remaining problem, with only a subset of the variables incorporated, is referred to as the master problem. Subsequently, variables that have the

Algorithm 2: Tabu Search and Row Generation algorithm

```
1  Construct BasicModel without crossing constraints and without prohibiting or enforcing
     matching constraints;
2  Set CurrentModel = BasicModel;
3  Initialize ProhibitedMatchings = ∅;
4  Initialize BlockingsCurrent = ∅;
5  Initialize BlockingsAll = ∅;
6  Set FeasibleSolution to false;
7  Set Basic to true;
8  while FeasibleSolution is false do
9      Find CurrentSolution by solving CurrentModel;
10     if No solution found in CurrentSolution then
11         if Basic = true then
12             No solution can be found overall;
13             break;
14         else
15             No solution can be found for current matching;
16             Set CurrentModel = BasicModel;
17             Set Basic = true;
18             Find matchings to prohibit based on blockings in previous iteration and store in
                 AddToProhibited;
19             Remove matchings from AddToProhibited if their inclusion prohibits all
                 matching possibilities for a certain train unit;
20             if AddToProhibited = ∅ then
21                 No solution can be found;
22                 break;
23             else
24                 Update ProhibitedMatchings with AddToProhibited;
25                 Generate constraints to prohibit matchings in ProhibitedMatchings and
                     update CurrentModel;
26             end
27         end
28     else
29         Find BlockingsCurrent in CurrentSolution;
30         if BlockingsCurrent = ∅ then
31             Set FeasibleSolution = true;
32         else
33             if Basic = true then
34                 Fix current matching and update CurrentModel;
35                 Set Basic = false;
36             end
37             Add crossing constraints current blockings to CurrentModel;
38             Add BlockingsCurrent to BlockingsAll
39         end
40     end
41 end
```

potential to improve the objective function value are generated and added to the master problem. Identifying appropriate variables with positive or negative (for a maximization or minimization problems, respectively) reduced costs to enter the basis is referred to as the subproblem. This procedure is repeated until no variables with positive reduced costs can be identified anymore. The power of column generation is that the subproblem, although often also NP hard, has a clear structure for which exact algorithms or dynamic programming algorithms exist, like shortest path or knapsack problems. Unfortunately, our subproblem lacks such a clear and easily identifiable structure. Also, it would be quite difficult to determine which subset of columns to start with in the master problem such that a solution to the master problem would also be a feasible solution to the original problem. With small adaptions, however, such as adding a variable z_t which is equal to 1 if train unit t is assigned to no track at all and equal to 0 if train unit t is assigned to a track, to the left hand side of constraints (4.21), and adding the z_t variables to the objective function with a large penalty $M \gg 0$, we could rather easily construct an appropriate initial master problem. This would entail only including the z_t variables in the basis to start with, which would be equal to one for all $t \in T$ in the optimal solution and would have an objective value of $M * |T|$. To illustrate, in the original problem the objective function (4.20) should be replaced by expression:

$$\sum_{(t,u)\in L} \gamma_{t,u} + \sum_{t\in T} M z_t \tag{5.1}$$

and constraints (4.21) should be replaced by:

$$\sum_{f\in F} x_{t,f} + z_t = 1 \quad \forall t \in T. \tag{5.2}$$

The initial master problem would become:

$$\{min\ \mathbf{Mz} | \mathbf{z} = \mathbf{1}, \mathbf{z} \in \mathbb{B}^{|\mathbf{T}|}\}, \tag{5.3}$$

where $\mathbf{M}$ would be a $1 \times |T|$ sized vector with at every location value M, $\mathbf{1}$ would be a $|T| \times 1$ sized vector consisting of only ones, and $\mathbf{z}$ would be a $|T| \times 1$ sized vector with variable z_t at index t for all $|T|$ indices. However, it would still be difficult to determine which variable could most beneficially enter the basis, since we cannot find an intuitively logical and clear structure for the subproblem.

For column generation to work, one could imagine it would be fruitful to rewrite the problem such that the variables would correspond to possible paths through the shunting yard for arriving train units for different matching possibilities. Paths should include the arrival and departure track of a train unit and its reallocation time if applicable. For each arriving train unit, only one path would be allowed to be chosen and additional constraints would be needed to ensure that for each departing train unit exactly one path is chosen as well, and constraints that would prevent combinations of routes that would result in crossings and combinations of routes that would result in exceeding track length capacity. These additional constraints, however, complicate the problem in such a way that the column generation subproblem is not a straightforward resource constrained shortest path problem anymore. Haahr et al. [14] already discussed the application of column generation for the TUSP without reallocation. They conclude that an exact application of column generation without heuristic procedures will result in very extensive and complicated networks and will likely be very slow.

As an alternative, we could work with 'half routes' for matchings. These 'half routes' would describe the path of an arriving train unit from its arrival until after its reallocation. For a departing train units, a 'half route' would describe its path from right after reallocation until its departure from the shunting yard. Then, we would have to link for each arriving unit one 'half route' to a departing 'half route' and vice versa. Although this resolves some of the problems we discussed, it poses new ones and therefore we decided to not explore this option any further. Considering the above, we pursued a row generation approach instead, which is a more logical and intuitive choice for a problem of our structure and with our characteristics.

Chapter 6

Data

We have two data sets with service schedules to which we can apply our solution methods to examine the results and practical applicability of our methods. Each data set applies to a different shunting yard: one applies to the shunting yard *Kleine Binckhorst* near The Hague Central Station and the other one to shunting yard *Utrecht OZ* near Utrecht Central Station. A map of the former is given in the introduction figure 1.2; a map of the latter is displayed in figure 6.2 at the end of this section.

For *Kleine Binckhorst*, the available track lengths are given in table 6.2. We are allowed to use all tracks with a non-black label as parking track, except tracks 51a, 64, and 104a, i.e. tracks 52-63. When solving the model for data sets that apply to *Kleine Binckhorst*, we merge tracks 62 and 63 to simplify the shunting yard lay out. Compositions arrive at and depart from the shunting yard via track 906a, but we assume they use track 51a to reach track 104a, and from there on they will be parked. When departing, they will follow the opposite route. Reallocation cannot take place via track 906a. This creates a simplified shunting yard structure with a diagonal track with LIFO tracks on both sides as discussed in subsection 4.3.2. A schematic overview of this structure can be found in figure 6.1 at the end of this section. Note that the orientation of this overview is upside down with respect to the map in figure 1.2. Since the moving direction is always changed at arrival and departure on track 104a, we assign tracks 52-59 to set F_N and tracks 60-62 to set F_D. This way, the ordering and constraints as defined in subsection 4.3.4 are respected.

For *Utrecht OZ*, the available track lengths are given in table 6.4 at the end of this section. We can use all tracks but 102, 103, 116, and 117 for parking, i.e. tracks 104-109 and tracks 115, 118, and 119. We merge tracks 105b and 105c and tracks 106b and 106c and use all tracks as LIFO tracks that can be reached through diagonal track 117, which is also the track via which train units arrive at and depart from the shunting yard. Again, the shunting yard consists of a diagonal track with LIFO tracks on both sides as discussed in subsection 4.3.2. A schematic overview of this new structure is displayed by the red lines in the map in figure 6.2. To apply the extensions discussed in subsection 4.3.4, we assign all tracks on the left of the diagonal track 117 to set F_N and all tracks on the right of the diagonal track 117 to set F_D.

The data sets contain all information regarding the services, the compositions linked to them, the train units in the compositions, and the type and number of carriages of these train units. The schedules are provided by *NS Reizigers*, and the task of *NedTrain* is to store the train units (and perform maintenance activities on them) and deliver them back to the main railway network in the correct compositions at the times as defined in the schedule. The data set for shunting yard *Kleine Binckhorst* contains 200 artificial instances. For each even number of arriving train units from 2 up until 40 -that is $2, 4, 6, \ldots, 38, 40$- ten artificial instances have been created that differ in service characteristics, so the compositions can be different as well as the arrival and departure times. The types and number of carriages of the train units used in the compositions and services are identical for each of the ten instances created for a certain number of arriving events, and the number of services is identical for each instance in an instance set as well. Naturally, the number

of train units in the planning problem is equal to two times the number of arriving train units, since we need as many arriving as departing train units. The planning period spans from 8:00 on one day till 8:00 the next day The OPG method as described in [20] and the methods as described in [3] have both been applied to this data set in [3], so we can compare the performance of our solution methods to the performance of their solution methods. An overview of the characteristics of the instance sets that apply to *Kleine Binckhorst* can be found in table 6.3 at the end of this section.

The data set for shunting yard *Utrecht OZ* contains the schedules for each day of a a typical week in December 2016. The schedules are modified such that the schedule again spans a 24-hour time period, from 8:00 on the specified day till 8:00 the next day. We have for each day the original schedule, and sometimes one or more schedules that are modified such that the number of carriages in the shunting yard at any given moment during the planning period never exceeds a certain number. The maximum number of carriages present in the shunting yard during the planning period can be lowered by adapting the arrival and departure times of the services in the schedule, or by adapting the number of carriages in the train units. This reduces track length capacity issues. For this data set, the matching that *NS Reizigers* prefers is included. An overview of the characteristics of the data set that applies to *Utrecht OZ* can be found in table 6.5. The first two letters of the instance name refer to the day of the week the instance represents, where 'MA', 'DI', 'WO', 'DO', 'VR', 'ZA', and 'ZO' refer to Monday to Sunday, respectively.

Table 6.1 displays the train unit lengths for the different types that appear in the data sets. Each row in the table refers to a unique type τ_t. In the data set for *Kleine Binckhorst*, types *VIRM*-4, *VIRM*-6, *SLT*-4, *SLT*-6, and *DDZ*-6 occur. In the data set for *Utrecht OZ*, all types but *DDZ*-6 occur.

Train Type	# Carriages	Length (m)	Train Type	# Carriages	Length (m)
VIRM	4	109	*SLT*	4	70
VIRM	6	162	*SLT*	6	101
DDZ	4	101	*ICMm*	3	81
DDZ	6	154	*ICMm*	4	107
SGMm	3	79	*DDAR*	4	97

Table 6.1: Train unit lengths

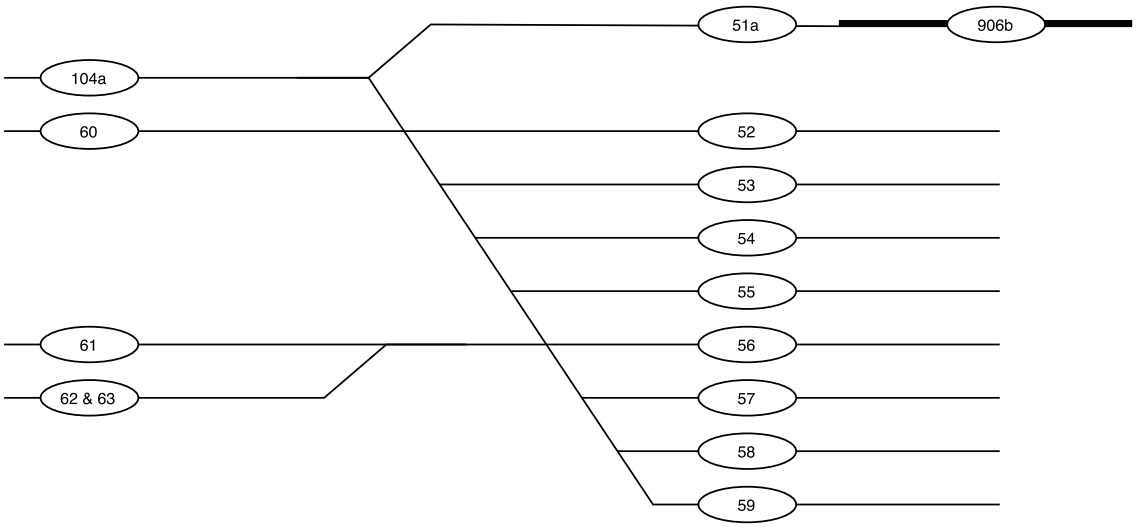


Figure 6.1: Simplified structure of Kleine Binckhorst

Track	Available length (m)
51a	62
52	480
53	431
54	387
55	357
56	222
57	202
58	203
59	183
60	248
61	247
62	247
63	326
64	220
104a	475

Table 6.2: Track lengths for shunting yard Kleine Binckhorst

Instance set	# Services	# Train units	Instance set	# Services	# Train units
2	4	4	22	30	44
4	4	8	24	32	48
6	8	12	26	35	52
8	11	16	28	37	56
10	14	20	30	40	60
12	17	24	32	44	64
14	18	28	34	44	68
16	20	32	36	48	72
18	23	36	38	51	76
20	26	40	40	54	80

Table 6.3: Characteristics of data set for shunting yard Kleine Binckhorst

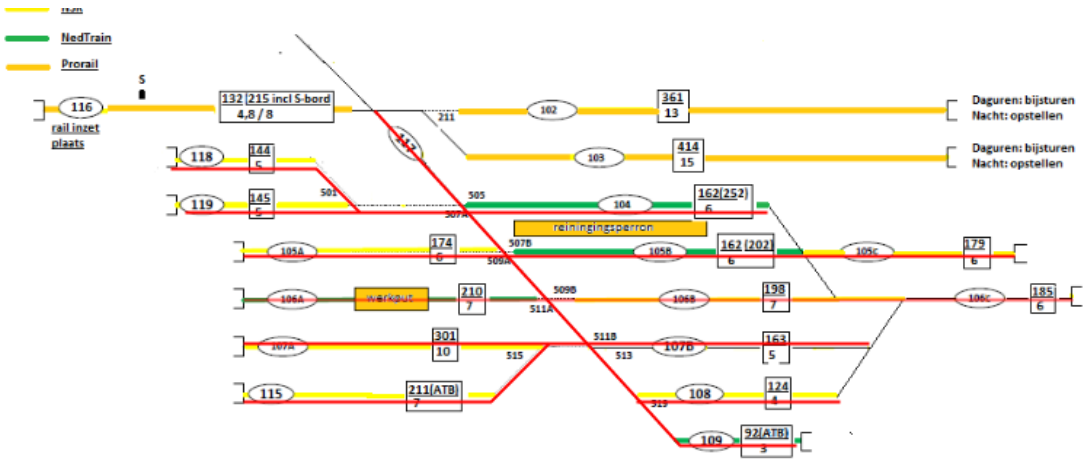


Figure 6.2: Map of Utrecht OZ

Track	Available length (m)
102	361
103	414
104	252
105a	174
105b	202
105c	179
106a	210
106b	198
106c	185
107a	301
107b	163
108	124
109	92
115	211
116	132
118	144
119	145

Table 6.4: Track lengths for shunting yard Utrecht OZ

Instance	# Services	# Train units	# Max carriages
MA_24_49	40	44	49
MA_24	40	44	
DI_24_44	44	50	44
DI_24	42	48	
WO_24	41	48	
DO_24_48	44	50	48
DO_24	43	50	
VR_24	31	34	
ZA_24	24	26	
ZO_24_40	24	28	40
ZO_24_46	31	36	46
ZO_24	30	36	

Table 6.5: Characteristics of data set for shunting yard Utrecht OZ

Chapter 7

Results

Since we have data sets that contain services with compositions consisting of multiple train units that apply to the shunting yards discussed in chapter 6, we will use the model extended with the couple and decouple constraints and the constraints for shunting yards with LIFO tracks on both sides of a diagonal track. That is, the model consists of constraints (4.20)-(4.29), (4.34)-(4.45), and (4.83)-(4.91). The full formulation is displayed in appendix B.

We will first discuss the results for the data set applied to *Utrecht OZ* in section 7.1. For this data set, we will show results for different settings of the parameters in the solution methods to evaluate the performance of the methods. Subsequently we will discuss the results for *Kleine Binckhorst* in section 7.2. We will compare the results to those of Van den Broek [3]. Note that we leave out his results for the instance set with two arriving events, since deliberation with the author learned that his results for this instance set are likely to be faulty: probably, the wrong problem instance set was used in his results. A comparison of the results for data set *Utrecht OZ* and *Kleine Binckhorst* is included in section 7.3.

All computational experiments in this chapter are performed on a computer with one Intel(R) Core(TM) i7-5500U CPU (2.39GHz) processor and 8 gigabytes of RAM. We use version 12.6 of the CPLEX solver to solve the MIPs that occur in our solution methods. A maximum computation time of 1800 seconds (30 minutes) is set for all experiments.

7.1 Results *Utrecht OZ*

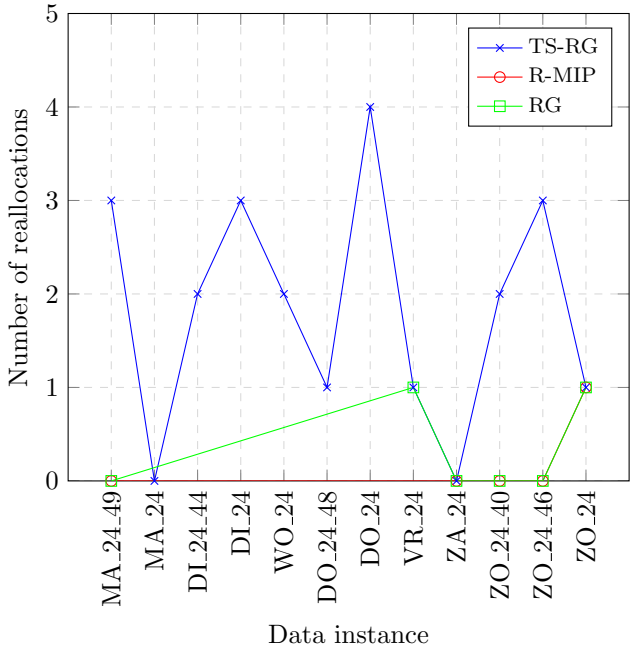
We applied four solution methods to the instances in data set *Utrecht OZ*. We solved the model for the TUSP as described in [19] (i.e. no reallocations allowed) for reference, we straightforwardly solved our model, we solved our model using row generation as described in algorithm 1, and we solved our model using the tabu search-row generation heuristic as described in algorithm 2. These solution methods are referred to as NR-MIP, R-MIP, RG, and TS-RG, respectively, from now on. In table 7.1 an overview of the results of applying these solution methods to our data instances is displayed. First of all, we see in the results for the reference NR-MIP method that three out of twelve data instances cannot be solved without using reallocation. The solution method is exact and we never exceed the maximum computation time, so failure to find a feasible solution in this

Method	Number of solved instances	Average solution time (s)	Average solution time if solved (s)
NR-MIP	9/12	2	1
R-MIP	5/12	1390	747
RG	6/12	1298	682
TS-RG	12/12	179	179

Table 7.1: Solved instances and computation times of different solution methods Utrecht OZ

Figure 7.1

Objective values R-MIP, RG, and TS-RG, *Utrecht OZ*



method means we must use reallocation to find a feasible solution. The computation times for this method are very short, since there are a lot less variables and constraints without reallocations incorporated in the model. The R-MIP and RG solution methods, which are both exact as well, find no solution in seven and six out of twelve instances, respectively. However, in the instances in which they failed to find a solution, they were cut short by the solution time limit and not necessarily by infeasibility. The RG method does result in slightly shorter computation times and since the number of instances solved is larger, it outperforms the R-MIP method. Experimenting with longer allowed computation times shows the R-MIP and RG solution methods can find the optimal solution if we allow the program to run for a longer period of time, however this is not useful for practical purposes. Finally, the TS-RG method finds a feasible solution in all twelve of the problem instances. The solution time of this method is a lot shorter compared to the R-MIP and RG methods, but a lot longer than the computation times of the reference NR-MIP method. However, as already mentioned the latter does not incorporate reallocation which reduces the dimensions of the constraint matrix greatly.

When we look at the objective value, i.e. the number of reallocations, of the instances for the R-MIP, RG, and TS-RG methods, we see that if they find a solution, the R-MIP and RG method always outperform the TS-RG method. Their objective value is always lower than the objective value when using TS-RG. This was to be expected, since the TS-RG method is a heuristic approach. An overview of the solution values is given in figure 7.1. If no solution is found, as is often the case for the R-MIP and RG methods, the data point is omitted.

Overall, the TS-RG method gives the best results in terms of computation times and number of solutions found. Since our problem is mainly a feasibility problem, we will continue with this solution method from now on, even though the objective values are not always as good as we like them to be.

As mentioned in chapter 6, *NS Reizigers* sometimes has a preferred matching for the train units. When we fix this matching, we expect the number of reallocations to increase, since we have less flexibility in our shunt plans. Also, the computation times will probably decrease, since the feasible region is considerably smaller. To reduce the computation time we can terminate CPLEX when

Figure 7.2

Computation times different settings TS-RG, *Utrecht OZ*

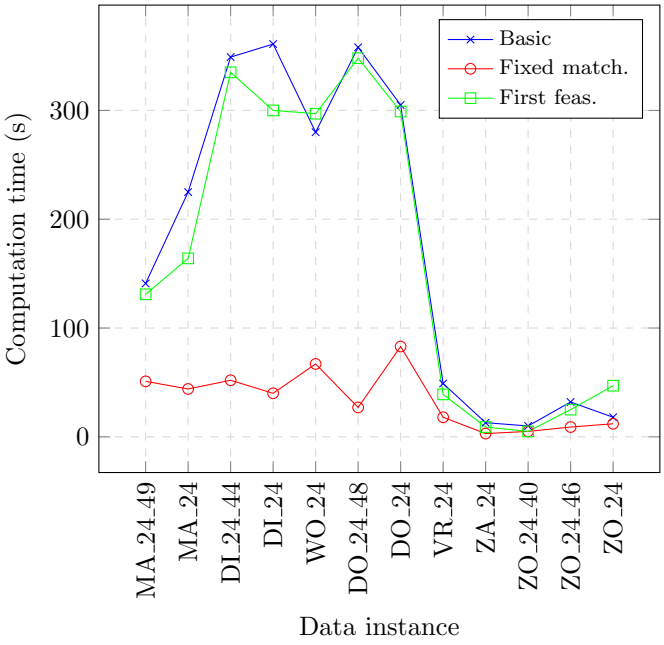
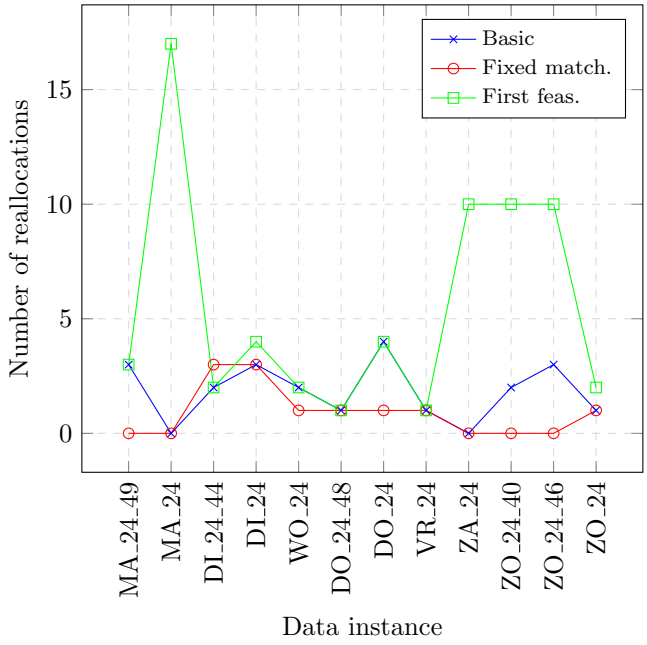


Figure 7.3

Objective values different settings TS-RG, *Utrecht OZ*



solving the MIPs as soon as a feasible solution has been found, as well. However, this is expected to increase the number of reallocations. The results of these experiments can be found in figures 7.2 and 7.3.

We see that the computation time when working with a single matching pattern is indeed a lot shorter, especially for the larger problem instances. However, the computation times when stopping after finding the first feasible solution and those when solving the MIPs in the TS-RG method to optimality are of the same order of magnitude, while the objective values of stopping after finding the first feasible solution are worse. Surprisingly, the objective values when solving the model for a predefined matching are smaller and thus better than those of the basic model. Since the first matching found using the TS-RG solution method always provided us with a feasible solution for all settings, we can draw the conclusion that this first matching is often not the best matching for our problem. Therefore, for larger problem instances as in *Kleine Binckhorst*, it might be interesting to fix the matching beforehand to reduce computation times.

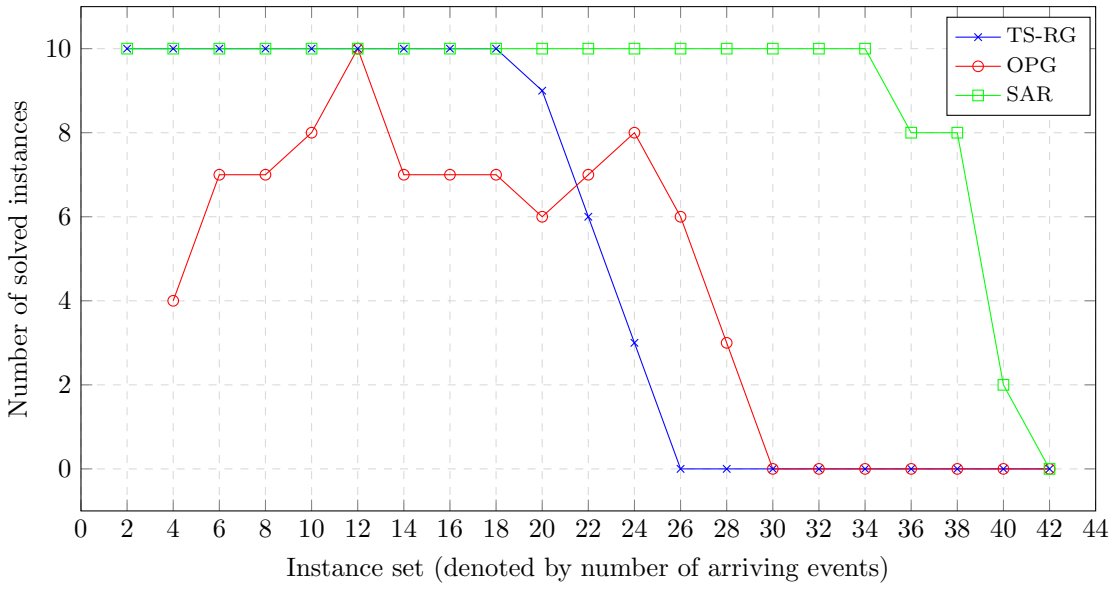
Since we always find a feasible solution for the first matching we use in the TS-RG heuristic, we find the same solution values when adjusting settings concerning the number of matchings to prohibit in the outer loop of algorithm 2, or when prohibiting only matchings that are blocked by others at departure from a track or alternatively prohibiting only matchings that are blocking others when attempting to depart from a track. When increasing the fixed moving times, multiple matchings are explored. However, in computational experiments we did not find any improvements when adapting the discussed settings and therefore there is no need to alter them.

7.2 Results *Kleine Binckhorst*

We now apply our TS-RG solution method to the data instances of *Kleine Binckhorst*. An overview of the number of instances solved by our TS-RG method compared to the number of instances solved by the OPG method and the simulated annealing algorithm with reallocation (SAR) as described in [3], which have been obtained by Van den Broek [3], is given in figure 7.4. In his results, the time limit was also set to 30 minutes, but since a different computer is used the results are not one on one comparable.

Figure 7.4

Performance OPG, SAR, and TS-RG, *Kleine Binckhorst*



Our method has a perfect record up until 18 arriving events. After that, the performance drops drastically. Zooming in on the results for our TS-RG method, we find that the method results in feasible solutions every time the solution time was not cut short by the solution time limit of 30 minutes. However, we run out of solution time for the first time for the instance set with 20 arriving events (thus 40 events in total) and from the instance set with 26 arriving events onward, we run out of solution time for every instance. This means that for these instances, our method might be able to find a feasible solution, but the computation times are simply too excessive. There is no time to fully explore all regions of the solution space. We can see that our method outperforms the OPG for smaller problem instances, but for larger problem instances (from 22 arriving events onward) it is outperformed by OPG. This is caused by the solution time limit. SAR outperforms our method. The results are the same up to 18 events, and after that the performance of the TS-RG drops. Again, the solution time required is the limiting factor. Note that although we expect the solution time to be the main limiting factor, our method is limited by the use of a simplified shunting yard layout as well, as explained in chapter 6. Since we do not fully exploit the possibilities the free tracks in shunting yard *Kleine Binckhorst* offer, we might not find a solution where there is one using the different routing possibilities free tracks offer. However, we cannot be sure of this, since we have never actually not found a solution: we ran out of computation time before this could be determined. The average computation times and solution values for the solved instances and the overall computation times using TS-RG are displayed in figure 7.5.

In the figure, the results for the average number of reallocations up to the instance set with 24 arriving units are shown, since from 26 arriving units onward, we always ran out of computation time when solving the model. The average number of reallocations is taken over the instances for which we found a solution. We can see from the figure that the average number of reallocations in the solutions increases with the problem size (the blue line), as do the computation times (the red and green lines). This is as we expected. The increase in computation times comes to a stop at around 1800 seconds, since this is the computation time limit. We also notice a sudden increase in computation times around the instance set with 20 arriving events. Therefore, 20 arriving events seems to be a breaking point in being able to feasibly solve the problem. This can also be seen in the number of instances solved in figure 7.4. We find that the results differ considerably from the results for instances with similar problem size in the *Utrecht OZ* data set. We will discuss possible

Figure 7.5

Average number of reallocations and computation time TS-RG, *Kleine Binckhorst*

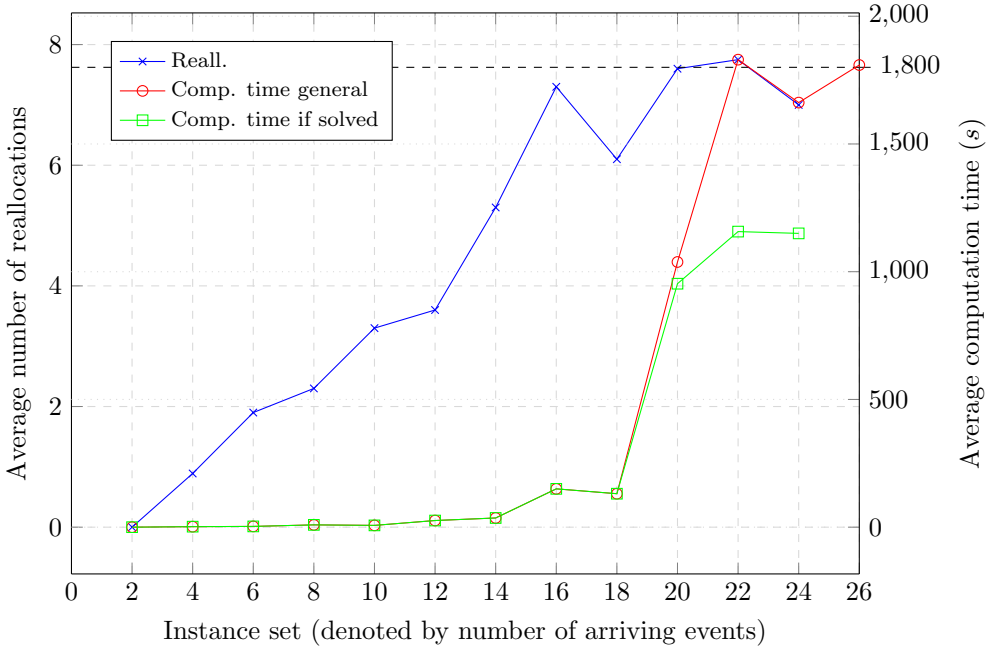
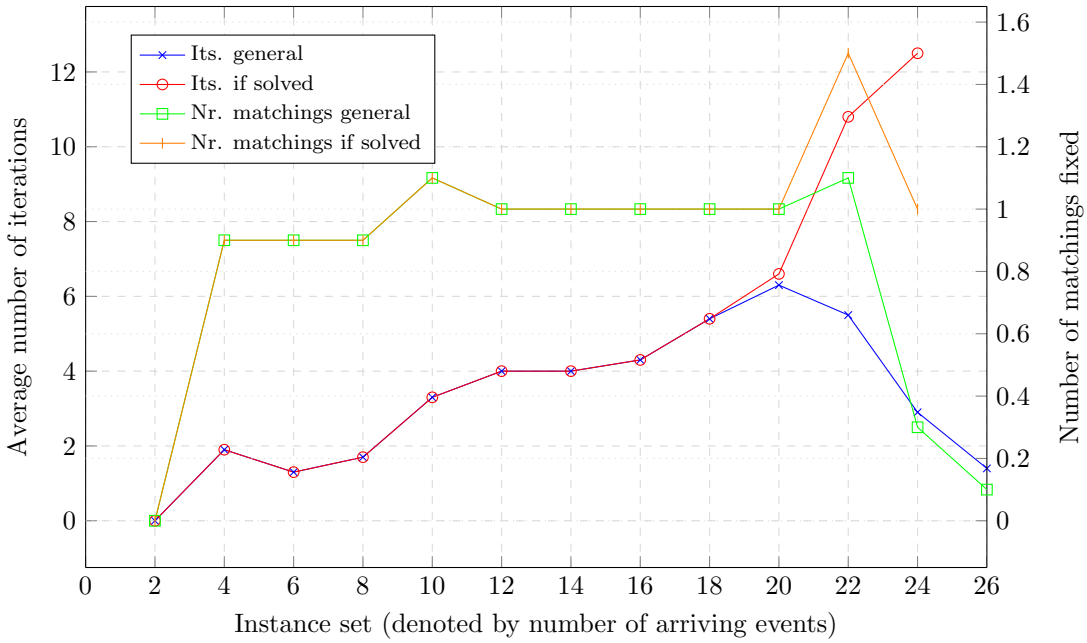


Figure 7.6

Average number of iterations TS-RG, *Kleine Binckhorst*



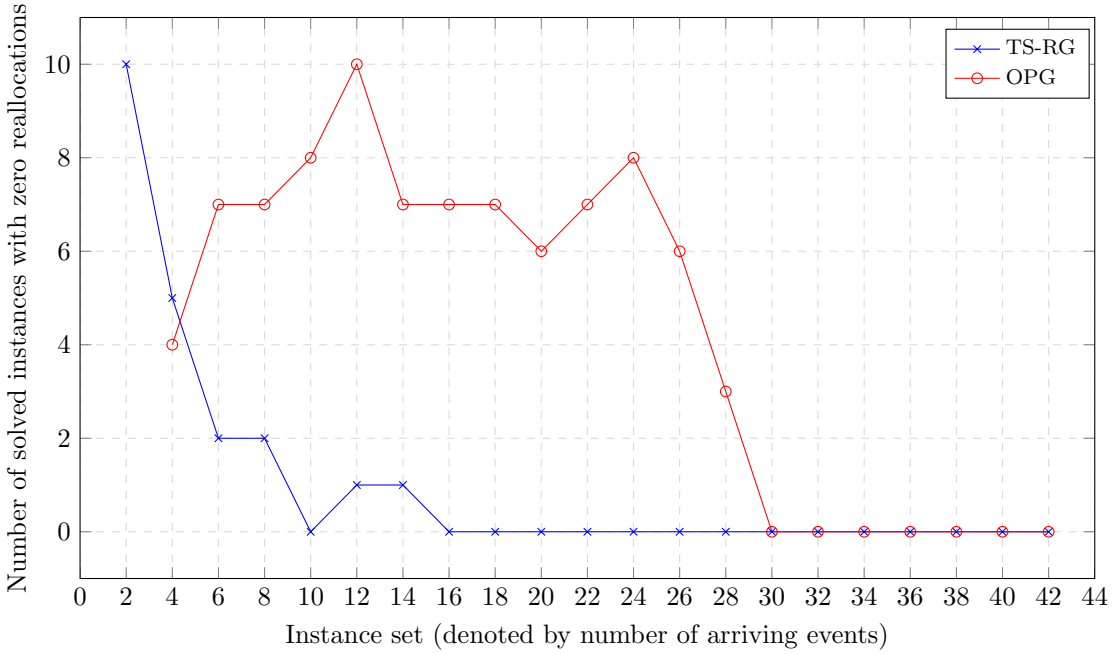
causes in section 7.3. A closer comparison of the average computation times for the instances that were solved versus the overall computation times shows big differences, especially for the larger instances where for some instances we did not find a feasible solution within our time limit. We can draw the conclusion that if we find a solution, we find it relatively fast. Therefore, we take a look at the average number of iterations our method uses in figure 7.6.

In figure 7.6, the number of iterations refers to the number of times we add crossing constraints to the current model, so the number of times we solve the model, find a solution, but this solution turns out to be infeasible. The number of matchings refers to the number of times we fix a different matching. We fix a different matching every time we cannot find a feasible solution for the current matching after adding crossing constraints. Therefore, an average of zero iterations combined with zero matchings refers to either the case where we instantly find a solution for the basic model in which no crossings occur, so there is no need to fix a matching or add any crossing constraints, or the case where we run out of solution time before we can even find a solution for the basic model. The former is generally the case for very small problem instances, the latter for larger instances. The number of iterations and the number of different matchings explored for the instances solved is larger than the general average for the instances where there is a difference. This leads us to believe that if we find a solution for the basic model, we can often go through multiple iterations and sometimes matchings to find a feasible solution relatively quickly. If we do not find a solution, this is often caused by inability to solve the basic model within the time limit. This is in line with the computation times in figure 7.5. Therefore, we will do some experiments with a predetermined, fixed matching later on in this section, so we can add more crossing constraints cuts and explore more regions of the solution space.

Since OPG does not allow for reallocations, all solutions the method finds are without reallocations. Therefore, we compare the number of solutions found without reallocations using TS-RG to the number of solutions found by OPG. The results are shown in figure 7.7. From figure 7.7, we can see that our method often does not find a solution with zero reallocations, while this solution does exist, as can be seen from the result for the OPG method. This can be explained by multiple factors. First of all, as we explained before, our method is a heuristic approach. For the fixed matching for which we find a solution, the solution is optimal. However, since we stop as soon

Figure 7.7

Number of solved instances with zero reallocations OPG and TS-RG, *Kleine Binckhorst*



as we find a solution without crossings for a matching, there might exist solutions without crossings with fewer reallocations for a different matching. Second, our method does not fully exploit the possibilities of the shunting yard layout of *Kleine Binckhorst* as already mentioned before. We model the shunting yard as if it has LIFO tracks, however, in actuality it also has many free tracks. Free tracks offer more routing possibilities, which likely results in fewer required reallocations. An exception occurs for the instance set with four arriving events. In this case, our method finds a solution with zero reallocations in five out of ten instances, where OPG finds one only in four out of ten. We cannot readily find an explanation for this aberrant result, so further research is needed to be able to correctly interpret it.

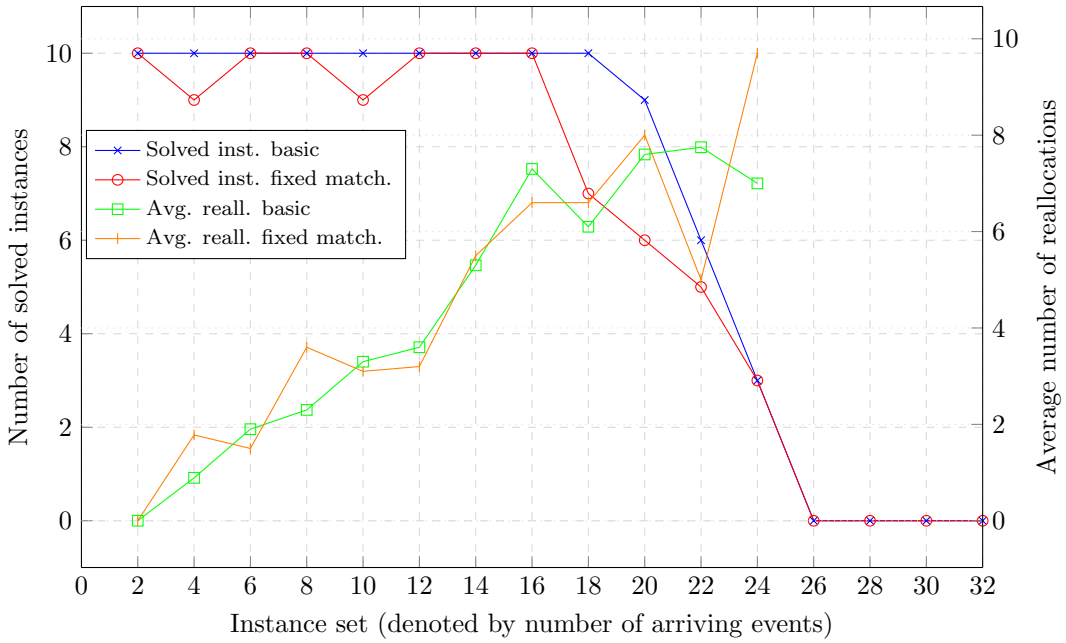
In the results for *Utrecht OZ* in section 7.1, we found that a fixed matching could help in reducing the computation times. In the results for the problem instances that did not run out of solution time for *Kleine Binckhorst*, we also find that the first matching found often results in a feasible solution. Therefore, we find for the problem instances a predefined matching by solving the Matching Problem as formulated in [9] before we apply our solution method. The results can be found in figure 7.8.

The figure shows similar results for the number of instances solved by the model with a predetermined matching and the basic model. The computation times were very similar as well. Most unsolved instances were caused by a lack of computation time, but the unsolved instances in the instance sets with 4, 10 and 18 arriving events were truly unsolved. Our predetermined matching was probably infeasible for these instances. The similar results for the number of solved instances when working with a fixed matching are in line with the results for the data set *Utrecht OZ*. However, we would expect a reduction in computation time as was the case for *Utrecht OZ*. Therefore, we expected more solved instances as well. We think this is caused by the 'randomness' of our predetermined matchings. The predetermined matching for *Utrecht OZ* was made by planners, who probably chose it smartly. The predetermined matching for *Utrecht OZ* also resulted in fewer reallocations, which we did not find in our results. This can again be explained by the difference in construction of the predetermined matching patterns.

These results strengthen our thoughts that the initial matching we choose might not be of main

Figure 7.8

Overview results different settings TS-RG, *Kleine Binckhorst*



importance for a feasible solution, but can be very important for a further reduction of the computation times. This can be very beneficial to improve the number of solved instances.

7.3 Comparison results *Utrecht OZ* and *Kleine Binckhorst*

We see that, compared to the results for instances applied to *Utrecht OZ* with comparable problem size, the solutions for *Kleine Binckhorst* are often characterized by more reallocations and longer computation times. Also, we run out of computation time for problem sizes that could be solved within our time limit for *Utrecht OZ*. Figure 7.9 and 7.10 show an overview of the average number of reallocations and average computation times of the two methods, respectively. The x-axis represents the number of arriving events in the problem instance. The average of the solution values for each instance size for each data set is given. Note that the results apply to different instance sets, meaning the instance set with 14 arriving events for *Utrecht OZ* is different than the instance set with 14 arriving events for *Kleine Binckhorst*. However, we do apply the instances for *Utrecht OZ* to the shunting yard layout of *Kleine Binckhorst* for better comparison. The data set for *Utrecht OZ* is a lot smaller than the data set for *Kleine Binckhorst*, therefore the averages of the solution values are taken over one to three instances, where they are taken over ten instances (or the number of solved instances) for *Kleine Binckhorst*. For data set *Kleine Binckhorst*, the figures also show the average number of reallocations and computation time when we ignore constraints (4.80), stating that train units that belong to the same composition should be parked on the same track. We will explain why in the further discussion of the results in figure 7.9 and 7.10.

We can explain the aberrant results in number of reallocations for comparable problem sizes in figure 7.9 by the difference in data characteristics: a quick comparison of table 6.3 and table 6.5 learns that in the problem instances for *Utrecht OZ*, the difference between the number of services and the number of train units is a lot smaller than for *Kleine Binckhorst*. Therefore, in the data set for *Utrecht OZ*, many more compositions exist of one train unit only, which means the constraint stating that train units belonging to the same composition must be parked on the same track often does not apply. This constraint increases the number of reallocations, since in practically all cases a reallocation can be prevented by assigning a different track to one of the train units involved

Figure 7.9

Average number of reallocations TS-RG, *Kleine Binckhorst* and *Utrecht OZ*

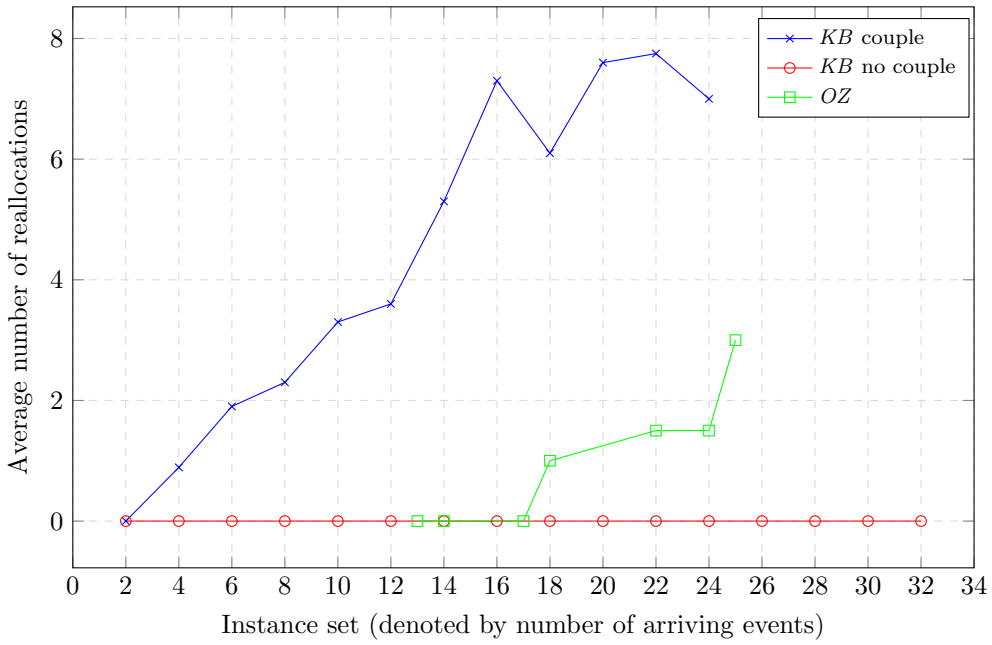
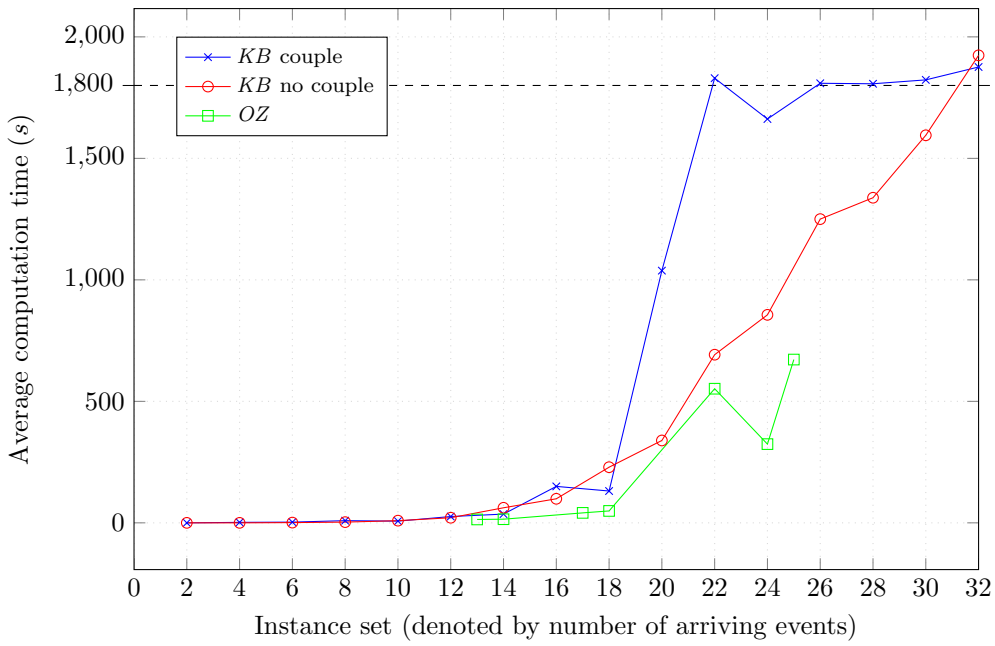


Figure 7.10

Average computation times TS-RG, *Kleine Binckhorst* and *Utrecht OZ*



in a crossing, as can be seen from the results for the model without including constraints (4.80) (the red line). It must be noted that we do not find a solution in all cases before we run out of computation time, so we might not be able to find a solution with zero reallocations for these instances, or any solution at all. When excluding constraints (4.80), we find a solution before we run out of computation time in slightly more instances as well.

The longer computation times can also partly be explained by this different number of services. Solving the model without constraints (4.80) can generally be done much faster as can be seen from the green line in figure 7.9. The longer computation times can also be caused by the difference in the number of matching patterns that exist for the different data sets. Since in the data set for *Utrecht OZ* many more different train types are used than in the data set for *Kleine Binckhorst* as discussed in chapter 6, the number of possible matching patterns is a lot smaller than for *Kleine Binckhorst*. Therefore, solving the basic model without crossing constraints and without having fixed a matching yet can be done much faster for instances in the data set for *Utrecht OZ*.

Chapter 8

Discussion

Several limitations apply to the model and our results as discussed in the previous chapters. First of all, the results for the OPG and SAR methods where obtained using different hard- and software than the results for TS-RG, which means the results are not one on one comparable. This is particularly relevant for the computation times and since the limited computation time played a role in all instances for which we did not find a feasible solution, this could influence our results and findings. When comparing results of the different methods it should also be noted that TS-RG does not fully exploit the shunting yard layout of *Kleine Binckhorst*. The yard consists partly of free tracks, which we use as LIFO tracks. This limits the routing possibilities and can thus increase the number of iterations and reduce the number of feasible solutions found. However, modelling free tracks enlarges the constraint matrix, which can increase computation times as well.

As mentioned in the introduction of chapter 4, our method does not take into account the order of reallocations in a time window. Crossings can occur in the particular case that two train units with the same arrival and departure track are reallocated at the same time. Only one reallocation order is possible (namely the last train unit to arrive on the arrival track should be reallocated first), while at departure from the departure track a different reallocation order could be required. We assume we can always feasibly order the reallocations in a time window, but this example shows this might not always be the case. As our formulation only considers one track at a time when assessing feasible routing, we do not prevent such cases in our formulation and thus our results could contain infeasibilities.

Furthermore, we assume a fixed moving time for all movements in a shunting yard, even though some routes can be more time consuming than others. We use the average moving time. It can therefore be the case that in some time windows, the cumulative moving time of all train units moved in this window is larger than the the actual time available, which makes our solution infeasible. On the other hand, when moving times in a time window are below average, the number of reallocations we allow in this window can be too limited, possibly resulting in cases where we cannot find a feasible solution even though one does exist. Besides this, we allow only one movement at a time in the shunting yard. However, in case their routes do not have overlapping segments, train units are allowed to move simultaneously in the yard. Incorporating this could increase the number of feasible solutions found.

Lastly, our routing only considers connections between tracks and available track length. Routing, however, is also dependent on the operation of switches. Operating switches takes time, and in some reallocation windows switches might have to be used in more than one setting. This might not always be doable within the given time constraints. Our model lacks the incorporation of switches and the time needed to change them. Therefore, some of our solutions might turn out to be infeasible.

Chapter 9

Conclusion

We formulated the TUSP-R as an MIP in which each train unit is allowed to be reallocated once during its stay in the shunting yard. Furthermore, multiple extensions to adapt the MIP to specific shunting yard layouts and data characteristics are provided. Since we are dealing with an $\mathcal{NP}$-hard problem, straightforwardly solving the MIP using CPLEX only results in optimal or even feasible solutions for very small problem instances. Although delayed row generation of the crossing constraints provided a small reduction of the computation times and increase of the number of solutions found for data set *Utrecht OZ*, the results were not satisfactory. When simply generating rows that correspond to crossings, the feasible region is still very large.

To more smartly search the solution space and to lower the computation times, a tabu search-row generation heuristic was proposed. In this heuristic, the problem is solved without the crossing constraints and when a solution is found, the current matching is fixed, which greatly reduces the feasible region of the problem. For this matching, we apply delayed row generation of the crossing constraints and if a feasible solution exists for the current matching pattern, the solution method will find it. If not, the fixed matching is discarded and elements of the matching pattern that were most often involved in crossings are prohibited when solving the model without crossing constraints again. This iteration will be faster, since the number of matching patterns in the solution space is reduced considerably.

The results of this tabu search-row generation heuristic are very good compared to straightforwardly solving the MIP and only applying row generation to the MIP. Computation times are reduced considerably and for real life problem instances as in data set *Utrecht OZ*, a feasible solution can always be found. However, this solution is not optimal, thus more reallocations than strictly needed will need to be performed. For the smaller artificial instances in *Kleine Binckhorst*, our method outperformed OPG and all problem instances were solved, as was the case for SAR as developed in [3]. However, for larger sized problem instances in this data set, the computation time always exceeded acceptable limits so the method was cut short and no solution was found. Interestingly, a solution could be found within time limits for similar sized instances in *Utrecht OZ*.

Closer investigation learned the excessive computation times were caused by the relatively large amount of time needed to calculate the first matching and track assignment when solving the model without crossing constraints. The subsequent iterations were considerably less time consuming with respect to this first step in the solution process. Instances for *Kleine Binckhorst* were characterized by relatively fewer services, meaning that more train units arrive as part of a composition. The requirement to park these units on the same track lead to less parking freedom and complexer problems. Furthermore, generally more reallocations are required to find a feasible solution which complicates and thus extends the solution process as well. Since it turned out that a feasible solution could often be found for the first matching pattern, a predefined matching was introduced to reduce the computation times. This predefined matching was obtained by solving the Matching Problem as formulated in [9] using CPLEX for the instances in *Kleine Binckhorst*; for the instances in *Utrecht OZ*, we were provided with a predetermined matching. The reduction

in computation times was visible for the *Utrecht OZ* instances, but not for the *Kleine Binckhorst* instances. We think this is caused by the difference in predefined matching that is used: we rather arbitrarily choose a matching for *Kleine Binckhorst*, where for *Utrecht OZ* the matching is probably more smartly chosen by the planners. Since the data set for *Kleine Binckhorst* generally contains instances with fewer different train types meaning more possible matching patterns exist, the likelihood that we choose an unfavorable one where a more favorable one exists is greater as well.

Concluding, we can say that including the possibility of reallocation increases the chance of being able to find a feasible shunt plan for a given problem instance. With the upcoming fleet expansion, *NedTrain* should definitely include this in their planning tools. For smaller shunting yards with smaller schedules, our model can be very useful in determining shunt plans; for larger instances, Van den Broek's SAR provides an outcome. For these instances, the tabu search-row generation approach does not reduce the computation time of our $\mathcal{NP}$-hard problem enough to be within acceptable ranges. Our method performs best for problem instances with diverse train types and relatively many different services on a LIFO style shunting yard layout. This keeps the size of the constraint matrix of the initial problem without crossing constraints relatively small compared to other types of problem instances.

9.1 Further Research

A real challenge proved to be modelling the specific shunting yard layouts and fully exploiting all of their routing possibilities. Shunting yards do not simply consist of LIFO or free tracks. Some structures do not provide for a connection between all tracks, unless some other parking track is empty. Therefore, our formulation and other exact formulations in literature simplify the shunting yard structure for general applicability of the models and for sake of simplicity. However, we believe efficient modelling of such complicated structures can lead to a reduction of the number of reallocations needed and an increase of the number of instances solved. Further research in smartly chosen formulations is therefore essential for further development of exact solution methods, especially when aiming to solve larger instances with relatively few services. For free tracks, the placement of train units on tracks is also an important planning factor as mentioned in section 4.3. Research should also be focused on finding smart ways to incorporate this element in the formulation.

Besides this, we think further development of our tabu search-row generation could benefit its performance. Smartly chosen matchings can potentially reduce the computation time of the basic model without crossing constraints and this allows for more iterations in the later steps of the solution approach. We expect the more iterations we can do, the more solutions can be found. On the other hand, it can also be interesting to switch between matching patterns sooner. If the current fixed matching is expected to be infeasible already after a couple of crossing constraints have been added, we might want to leave the current region of the solution space before having fully searched this region. In our current results, for example, a matching pattern often turns out to be infeasible if more and more crossings occur in the solution of each iteration, thus with every set of crossing constraints we add to the model. When the number of crossings goes down as the number of iterations goes up for a particular matching, we often do find a solution. However, these are preliminary findings and further experiments should be conducted for verification.

Next, we assume that compositions consisting of multiple units are decoupled right after their arrival at the shunting yard or coupled right before their departure from the yard. However, it might be smart to couple or decouple at a different time, or if possible not couple and decouple at all. The latter could be the case for train units both arriving and departing in the same service, which means they can stay together throughout their entire stay in the shunting yard. This could be very beneficial, since fewer reallocations would need to be performed since these train units can be reallocated simultaneously as one composition. It must however be noted that this further complicates the model as it would increase the number of variables and constraints.

To fully grasp the potential of our method compared to existing methods like OPG and SAR,

their performance should be evaluated for data set *Utrecht OZ* as well. Our simplified shunting yard layout for *Utrecht OZ* was much more similar to the actual layout than was the case for *Kleine Binckhorst*, meaning the results of the different methods can better be compared. We did not include this in our research as the data were not readily available, but it can rather easily be tested.

Lastly, it would naturally be very interesting to formulate the model such that multiple reallocations per train unit are allowed. As stated before, we do not expect many train units to be reallocated multiple times because of the limited time available for reallocations, but maybe one or two repeatedly reallocated train units could improve the number of feasible instances. We do not expect that in total many more reallocations would be executed in this option. Perhaps even less, as it could lead to a different distribution of reallocations over the train units. Certain matching elements could be reallocated multiple times, where others would not be reallocated at all. However, further research is needed to verify this hypothesis. Please note that incorporating multiple reallocations would increase the dimensions of the constraint matrix considerably. Therefore, we expect even more excessive computation times when applying the solution methods in this thesis to a model which allows for multiple reallocations per train unit, especially for larger sized problem instances.

Bibliography

[1] Benders, J. F. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik 4*, 1 (1962), 238–252.

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

[3] Broek, R. W. van den. Train shunting and service scheduling: an integrated local search approach. Master's thesis, Utrecht University, 2016.

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

[5] DePuy, G. W., and Taylor, G. D. Using board puzzles to teach operations research. *INFORMS Transactions on Education 7*, 2 (2007), 160–171.

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

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

[8] Flake, G. W., and Baum, E. B. Rush Hour is PSPACE-complete, or "Why you should generously tip parking lot attendants". *Theoretical Computer Science 270*, 1 (2002), 895–911.

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

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

[11] Glover, F. Future paths for integer programming and links to artificial intelligence. *Computers & Operations Research 13*, 5 (1986), 533–549.

[12] Guignard, M., and Kim, S. Lagrangean decomposition: A model yielding stronger Lagrangean bounds. *Mathematical Programming 39*, 2 (1987), 215–228.

[13] Haahr, J. T., Lusby, R. M., Larsen, J., and Pisinger, D. Simultaneously recovering rolling stock schedules and depot plans under disruption. In *13th Conference on Advanced Systems in Public Transport* (2015).

[14] Haahr, J. T., Lusby, R. M., and Wagenaar, J. C. A comparison of optimization methods for solving the depot matching and parking problem. ERIM Report Series Research in Management ERS-2015-013-LIS, Erasmus Research Institute of Management, 2015.

[15] Hartog, M. Shunt planning: an integral approach of matching, parking and routing. Master's thesis, University of Twente, 2010.

[16] Hearn, R. A. The complexity of sliding-block puzzles and plank puzzles. In *Tribute to a Mathemagician*. A K Peters, 2004, pp. 173–183.

[17] Hearn, R. A., and Demaine, E. D. The nondeterministic constraint logic model of computation: reductions and applications. In *International Colloquium on Automata, Languages, and Programming*. Springer, 2002, pp. 401–413.

[18] Kaufman, L., and Broeckx, F. An algorithm for the quadratic assignment problem using Bender's decomposition. *European Journal of Operational Research 2*, 3 (1978), 207–211.

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

[20] Lentink, R. M. *Algorithmic decision support for shunt planning*. PhD thesis, Erasmus University Rotterdam, 2006.

[21] Lentink, R. M., Fioole, P.-J., Kroon, L. G., and Woudt, C. van 't. Applying operations research techniques to planning of train shunting. In *Planning in Intelligent Systems*. John Wiley Sons, Inc., 2006, pp. 415–436.

[22] Mitchell, J. E. Branch-and-cut algorithms for combinatorial optimization problems. In *Handbook of Applied Optimization*. Oxford University Press, 2000.

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

[24] Rijn, J. N. van. Rush Hour is PSPACE-complete. Tech. rep., Leiden Institute of Advanced Computer Science, 2011.

[25] Winter, T., and Zimmermann, U. T. Real-time dispatch of trams in storage yards. *Annals of Operations Research 96*, 1-4 (2000), 287–315.

Appendix A

Notation

In this appendix, we give an overview of the decision variables that are used in the different formulations. Sets and parameters that do not occur in the overview in tables 3.1 and 3.2 are also displayed.

A.1 Overview variables initial formulation

The following variables and definitions are used in the formulation in section 4.1:

- $x_{t,f} = \begin{cases} 1 & \text{if train unit } t \in T_+ \text{ is is assigned to track } f \in F \text{ (so } x_t = 1\text{)}, \\ 0 & \text{otherwise;} \end{cases}$
- $x_{t,u,f,g} = \begin{cases} 1 & \text{if arriving train unit } t \in T_+ \text{ is matched to departing train unit } u \in T_- \\ & \text{with } (t,u) \in L \text{ and the arrival track is } f \in F \text{ and the departure track is} \\ & g \in F \text{ (so } x_{t,f} = 1 \text{ and } x_{u,g} = 1\text{)}, \\ 0 & \text{otherwise;} \end{cases}$
- $y_{t,u,f,g,z} = \begin{cases} 1 & \text{if arriving train unit } t \in T_+ \text{ is matched to departing train unit } u \in T_- \\ & \text{with } (t,u) \in L, \text{ the arrival track is } f \in F \text{ and the departure track is } g \in F \\ & \text{(so } x_{t,f} = 1 \text{ and } x_{u,g} = 1\text{), and is reallocated between event } z-1 \text{ and } z, \\ & z \in T \\ 0 & \text{otherwise;} \end{cases}$
- $b_{f,t}$ cumulative length of train units on track $f \in F$ right after event $t \in T$;
- $\alpha_{t,u,f}$ arrival time of arriving train unit t matched to departing train unit u, $(t,u) \in L$, on track $f \in F$;
- $\delta_{t,u,f}$ departure time of arriving train unit t matched to departing train unit u, $(t,u) \in L$, from track $f \in F$.

A.2 Overview variables final formulation

The following variables are used in the formulation in section 4.2:

- $x_{t,f} = \begin{cases} 1 & \text{if train unit } t \in T_+ \text{ is is assigned to track } f \in F \text{ (so } x_{t,f} = 1\text{)}, \\ 0 & \text{otherwise;} \end{cases}$
- $x^1_{t,u,f} = \begin{cases} 1 & \text{if arriving train unit } t \in T_+ \text{ is matched to departing train unit } u \in T_- \\ & \text{and the arrival track is } f \in F \text{ (so } x_t = 1\text{)}, \\ 0 & \text{otherwise;} \end{cases}$

- $x^2_{t,u,f}$ $= \begin{cases} 1 & \text{if arriving train unit } t \in T_+ \text{ is matched to departing train unit } u \in T_- \\ & \text{and the departure track is } f \in F \text{ (so } x_{u,f} = 1), \\ 0 & \text{otherwise;} \end{cases}$

- $m_{t,u,z}$ $= \begin{cases} 1 & \text{if arriving train unit } t \in T_+ \text{ is matched to departing train unit } u \in T_- \\ & \text{and moved to its departure track between event } z-1 \text{ and } z,^1 \\ 0 & \text{otherwise;} \end{cases}$

- $\gamma_{t,u}$ $= \begin{cases} 1 & \text{if train unit } t \text{ matched to train unit } u,\ (t,u) \in L \text{ reallocates during its} \\ & \text{stay in the shunting yard,} \\ 0 & \text{otherwise;} \end{cases}$

- $\gamma_{t,u,f}$ $= \begin{cases} 1 & \text{if train unit } t \text{ matched to train unit } u,\ (t,u) \in L \text{ reallocates from or to} \\ & \text{track } f \text{ during its stay in the shunting yard,} \\ 0 & \text{otherwise;} \end{cases}$

- $b_{f,t}$ cumulative length of train units on track $f \in F$ right after event $t \in T$.

A.3 Overview notation extensions

The following variables, definitions, parameters, and sets are used in the problem extensions in section 4.3:

- $\alpha_{t,u,f}$ arrival time of arriving train unit t matched to departing train unit u, $(t,u) \in L$, on track $f \in F$;

- $\delta_{t,u,f}$ departure time of arriving train unit t matched to departing train unit u, $(t,u) \in L$, from track $f \in F$;

- ϕ_t $= \begin{cases} 0 & \text{if train unit } t \in T \text{ arrives or departs via the A-side,} \\ 1 & \text{if train unit } t \in T \text{ arrives or departs via the B-side;} \end{cases}$

- $\psi^1_{t,u} = \begin{cases} 0 & \text{if train unit } t \in T_+ \text{ matched to train unit } u \in T_- \text{ departs via the A-side} \\ & \text{during reallocation,} \\ 1 & \text{if train unit } t \in T_+ \text{ matched to train unit } u \in T_- \text{ departs via the B-side} \\ & \text{during reallocation;} \end{cases}$

- $\psi^2_{t,u} = \begin{cases} 0 & \text{if train unit } t \in T_+ \text{ matched to train unit } u \in T_- \text{ arrives via the A-side} \\ & \text{during reallocation,} \\ 1 & \text{if train unit } t \in T_+ \text{ matched to train unit } u \in T_- \text{ arrives via the B-side} \\ & \text{during reallocation;} \end{cases}$

- q_f maximum length for a train unit to be assigned to track $f \in F$;

- $n^*_{f,g,z}$ maximum length for a train unit reallocating between track f and g, $f,g \in F$ between event z and $z-1$, $z \in T$;

- O the set of all pairs of tracks $f,g \in F$ with a fixed maximum train unit length to reallocate from track f to g;

- J the set that contains all routes between all possible pairs of tracks f and g such that $(f,g) \notin O$;

- $J_{f,g}$ the set that contains all routes between track f and g such that $(f,g) \notin O$, $J_{f,g} \subset J$;

- F_j the set that contains for a route $j \in J$ the parking tracks $h \in F$ that are used in this route, $F_j \subset F$;

[1] Note that $m_{t,u,t} = 1$ implies that a train unit is never reallocated: it is moved to its departure track at time t, which means the arrival track and departure track are the same.

- $d_{j,h}$ the length of the part of the train unit that does not need to enter track $h \in F$ when reallocating via route $j \in J$;
- $e_j = \begin{cases} n_{f,g} & \text{if route } j \in J_{f,g} \text{ does not use any parking tracks when reallocating, but only uses the diagonal track as in the original formulation,} \\ 0 & \text{otherwise;} \end{cases}$
- $r_{j,z}$ the maximum allowed reallocation length of route $j \in J$ when reallocating between event $z-1$ and z, $z \in T$;
- $s_{j,z} = \begin{cases} n_{f,g} & \text{if the maximum train unit reallocation length for a time } z \in T \text{ is equal to at least } r_{j,z},\ j \in J, \\ 0 & \text{otherwise;} \end{cases}$
- a_t the train service in which train unit $t \in T$ arrives at or departs from the shunting yard;
- A set of pairs of train units t and $t+1$, $(t, t+1) \in T^2$, that arrive or depart in the same train service, that is $a_t = a_{t+1}$;
- T^c set of train units that arrive or depart in a composition of multiple train units;
- T^c_+ set of arriving train units that arrive in a composition of multiple train units;
- T^c_- set of departing train units that depart in a composition of multiple train units;
- χ_t the earliest allowed reallocation time for arriving train units, so $t \in T^c_+$, and the latest allowed reallocation time for departing train units, so $t \in T^c_-$;
- c_t the length of the composition train unit $t \in T$ is a part of;
- F_D set of tracks that are directly accessible from the diagonal track, $F_D \subset F$;
- F_N set of tracks that are not directly accessible from the diagonal track, $F_D \subset F$.

Appendix B

MIP formulation used for obtaining the computational results

The linearized version of the MIP formulation in this section is used to obtain the computational results in this thesis.

Minimize:

$$\sum_{(t,u)\in L} \gamma_{t,u}$$

Subject to:

$$\begin{aligned}
\sum_{f\in F} x_{t,f} &= 1 && \forall t \in T \\
\sum_{\substack{u: \\ (t,u)\in L}} x^1_{t,u,f} &= x_{t,f} && \forall t \in T_+,\ f \in F \\
\sum_{\substack{t: \\ (t,u)\in L}} x^2_{t,u,f} &= x_{u,f} && \forall u \in T_-,\ f \in F \\
\sum_{f\in F} x^1_{t,u,f} &= \sum_{f\in F} x^2_{t,u,f} && \forall (t,u) \in L \\
\gamma_{t,u,f} &= \left| x^1_{t,u,f} - x^2_{t,u,f} \right| && \forall (t,u) \in L,\ f \in F \\
\sum_{f\in F} \gamma_{t,u,f} &\leq 2\gamma_{t,u} && \forall (t,u) \in L \\
\gamma_{t,u} &\leq \frac{1}{2} \sum_{f\in F} \gamma_{t,u,f} && \forall (t,u) \in L \\
\sum_{\substack{z\in T \\ t<z\leq u}} m_{t,u,z} &= \gamma_{t,u} && \forall (t,u) \in L \\
\sum_{\substack{z\in T \\ t\leq z\leq u}} m_{t,u,z} &= \sum_{f\in F} x^1_{t,u,f} && \forall (t,u) \in L
\end{aligned}$$

$$m_{t,u,z}x_{u,f} + x_{t',f} - \sum_{\substack{u'\in T_-:\\(t',u')\in L,\\u'<u}} \sum_{\substack{z'\in T:\\t'\le z'\le u'}} m_{t',u',z'}$$
$$- \sum_{\substack{u'\in T_-:\\(t',u')\in L,\\u'>u}} \sum_{\substack{z'\in T:\\t'<z'\le u}} m_{t',u',z'} \le 1 \qquad \forall (t,u)\in L,\ z\in T: t\le z\le u,$$
$$t'\in T_+: z\le t'<u, t'\ne t,\ f\in F_D$$

$$m_{t,u,z}x_{u,f} + x_{t',f} - \sum_{\substack{u'\in T_-:\\(t',u')\in L,\\u'<^* u}} \sum_{\substack{z'\in T:\\t'\le^* z'\le^* u'}} m_{t',u',z'}$$
$$- \sum_{\substack{u'\in T_-:\\(t',u')\in L,\\u'>^* u}} \sum_{\substack{z'\in T:\\t'<^* z'\le^* u}} m_{t',u',z'} \le 1 \qquad \forall (t,u)\in L,\ z\in T: t\le z\le u,$$
$$t'\in T_+: z\le^* t'<^* u, t'\ne t,\ f\in F_N$$

$$m_{t,u,z}x_{u,f} + x_{u',f} - \sum_{\substack{t'\in T_+:\\(t',u')\in L,\\t'>u}} \sum_{\substack{z'\in T:\\t'\le z'\le u'}} m_{t',u',z'}$$
$$- \sum_{\substack{t'\in T_+:\\(t',u')\in L,\\t'<u}} \sum_{\substack{z'\in T:\\t'\le z'\le z\\u<z'\le u'}} m_{t',u',z'} \le 1 \qquad \forall (t,u)\in L,\ z\in T: t\le z\le u,$$
$$u'\in T_-: u'>u,\ f\in F_D$$

$$m_{t,u,z}x_{u,f} + x_{u',f} - \sum_{\substack{t'\in T_+:\\(t',u')\in L,\\t'>^* u}} \sum_{\substack{z'\in T:\\t'\le^* z'\le^* u'}} m_{t',u',z'}$$
$$- \sum_{\substack{t'\in T_+:\\(t',u')\in L,\\t'<^* u}} \sum_{\substack{z'\in T:\\t'\le^* z'\le^* z\\u<^* z'\le^* u'}} m_{t',u',z'} \le 1 \qquad \forall (t,u)\in L,\ z\in T: t\le z\le u,$$
$$u'\in T_-: u'>^* u,\ f\in F_N$$

$$m_{t,u,z}x_{t,f} + x_{t',f} - \sum_{\substack{u'\in T_-:\\(t',u')\in L,\\u'<z}} \sum_{\substack{z'\in T:\\t'\le z'\le u'}} m_{t',u',z'}$$
$$- \sum_{\substack{u'\in T_-:\\(t',u')\in L,\\u'\ge z}} \sum_{\substack{z'\in T:\\t'<z'\le z}} m_{t',u',z'} \le 1 \qquad \forall (t,u)\in L,\ z\in T: t< z\le u,$$
$$t'\in T_+: t<t'<z,\ f\in F_D$$

$$m_{t,u,z}x_{t,f} + x_{t',f} - \sum_{\substack{u'\in T_-:\\(t',u')\in L,\\u'<^* z}} \sum_{\substack{z'\in T:\\t'\le^* z'\le^* u'}} m_{t',u',z'}$$
$$- \sum_{\substack{u'\in T_-:\\(t',u')\in L,\\u'\ge^* z}} \sum_{\substack{z'\in T:\\t'<^* z'\le^* z}} m_{t',u',z'} \le 1 \qquad \forall (t,u)\in L,\ z\in T: t< z\le u,$$
$$t'\in T_+: t<^* t'<^* z,\ f\in F_N$$

$$m_{t,u,z}x_{t,f} + x_{u',f} - \sum_{\substack{t'\in T_+:\\(t',u')\in L,\\t'\ge z}} \sum_{\substack{z'\in T:\\t'\le z'\le u'}} m_{t',u',z'}$$
$$- \sum_{\substack{t'\in T_+:\\(t',u')\in L,\\t'<z}} \sum_{\substack{z'\in T:\\t'\le z'\le t\\z\le z'\le u'}} m_{t',u',z'} \le 1 \qquad \forall (t,u)\in L,\ z\in T: t< z\le u,$$
$$u'\in T_-: u'>z,\ f\in F_D$$

$$m_{t,u,z}x_{t,f} + x_{u',f} - \sum_{\substack{t'\in T_+:\\(t',u')\in L,\\ t'\geq^* z}} \sum_{\substack{z'\in T:\\ t'\leq^* z'\leq^* u'}} m_{t',u',z'}$$

$$- \sum_{\substack{t'\in T_+:\\(t',u')\in L,\\ t'<^* z}} \sum_{\substack{z'\in T:\\ t'\leq^* z'\leq^* t\\ z\leq^* z'\leq^* u'}} m_{t',u',z'} \leq 1 \qquad \forall (t,u)\in L,\ z\in T: t<z\leq u,\ u'\in T_-: u'>^* z,\ f\in F_N$$

$$b_{f,0} = 0 \qquad \forall f\in F$$

$$b_{f,t-1} + v_t k_t x_{t,f} + \sum_{\substack{(t',u')\in L:\\ t'<t\leq u'}} x_{u',f} m_{t',u',t} k_t$$

$$- \sum_{\substack{(t',u')\in L:\\ t'<t\leq u'}} x_{t',f} m_{t',u',t} k_t = b_{f,t} \qquad \forall f\in F,\ t\in T$$

$$b_{f,t} \leq l_f \qquad \forall t\in T,\ f\in F$$

$$x^1_{t,u,f} x_{u,g} k_t \leq n_{f,g} \qquad \forall (t,u)\in L,\ f,g\in F$$

$$\sum_{\substack{(t,u)\in L:\\ t<z\leq u}} m_{t,u,z} \leq w_z \qquad \forall z\in T$$

$$\sum_{\substack{u\in T_-:\\(t,u)\in L}} \sum_{\substack{z\in T:\\ t<z<\chi_t}} m_{t,u,z} = 0 \qquad \forall t\in T^c_+$$

$$\sum_{\substack{t\in T_+:\\(t,u)\in L}} \sum_{\substack{z\in T:\\ \chi_t<z\leq u}} m_{t,u,z} = 0 \qquad \forall u\in T^c_-$$

$$c_t x_{t,f} \leq q_f \qquad \forall f\in F$$

$$x_{t,f} \in \{0,1\} \qquad \forall t\in T,\ f\in F$$

$$x^1_{t,u,f} \in \{0,1\} \qquad \forall (t,u)\in L,\ f\in F$$

$$x^2_{t,u,f} \in \{0,1\} \qquad \forall (t,u)\in L,\ f\in F$$

$$m_{t,u,z} \in \{0,1\} \qquad \forall t,u,z\in T$$

$$\gamma_{t,u,f} \in \{0,1\} \qquad \forall (t,u)\in L,\ f\in F$$

$$\gamma_{t,u} \in \{0,1\} \qquad \forall (t,u)\in L$$

$$b_{f,t} \geq 0 \qquad \forall f\in F,\ t\in T$$