Master thesis

Heuristics for strategic passenger-oriented train timetabling

Timo den Dulk
(4954017)

supervised by:
Leo van Iersel (TU Delft)
Gabor Maroti (NS)
September 16, 2024

Abstract

This thesis addresses a timetabling problem known as Strategic Passenger-Oriented Timetabling (SPOT). SPOT is a timetabling problem that involves assigning departure and arrival times to train services. Unlike classical timetabling, which typically emphasizes infrastructural feasibility, SPOT shifts the focus toward enhancing the passenger experience without infrastructural constraints. The timetables generated by SPOT provide information regarding important connections for passengers and can later be adjusted to accommodate infrastructural constraints. SPOT evaluates a timetable based on the perceived travel time consisting of drive, dwell and transfer times, along with additional penalties for inconveniences by transfers and initial waiting times of passengers.

An initial research focused on using a Mixed-Integer Linear Programming (MILP) approach to solve the SPOT problem. However, this MILP formulation has multiple problems. The MILP tends to perform slow for small problem instances and becomes significantly worse as the problem size increases. Additionally, the effectiveness of the timetables generated by the MILP are often unreliable, because there is a substantial gap between the perceived travel time of the best-found timetable and the best-found lower bound.

This thesis explores a heuristic approach to solving SPOT. Specifically, two heuristics are employed: Local Search and Simulated Annealing. Both operate by exploring a neighborhood of possible timetables around a current timetable to identify better alternatives. The absence of infrastructural constraints in SPOT simplifies the creation of these neighborhoods, as different train services do not influence each other. Additionally, multiple lower bounds are calculated for the perceived travel time, providing benchmarks to evaluate the quality of the solutions. Overall, these heuristics are effective in finding good timetables within a short period, with the lower bounds being reasonably close to the perceived travel time of the generated timetables.

Contents

Chapter 1

Introduction

In 2023, 1.1 billion journeys were made by public transport in the Netherlands [10]. A large proportion of these journeys were by train, with NS (Nederlandse Spoorwegen, the Dutch Railways) being the largest provider. In 2023, NS facilitated an average of 1,085,000 train journeys per business day in the Netherlands. At the beginning of 2024, NS operateted 777 rolling stock units (train units) with a total seating capacity of 246,616. NS uses the HRN (hoofdrailnet) railway network, for which it has exclusive rights for passenger transport. Besides NS, other companies also operate on the Dutch railway network. For passenger transport, regional train providers include Arriva, Connexxion, and Keolis and for freight transport, DB Cargo (a subsidiary of the German railway company Deutsche Bahn) is the largest provider.

For this crowded railway network, NS needs to design a timetable of train services that can be executed safely, efficiently, and effectively to provide fast connections between stations. Such a timetable includes the departure and arrival times at which trains arrive and depart from stations. This timetable requires constant updates to accommodate the changing train network. For example, the number of passengers increases almost every year, requiring additional train services to meet the growing demand. In addition, changes to the infrastructure of stations and the railway network should be reflected in new timetables. Overall, this results in a very complex problem that involves multiple planning phases, some of which already start years in advance.

1.1 Railway planning process at NS

The process of creating a timetable containing departure and arrival times is called timetabling, which is just one problem of the overall railway planning processes. First, several other planning problems must be addressed to provide the necessary input for timetabling. Additionally, once a timetable is created, various planning problems must be solved to ensure that the timetable is safe and efficient to implement. To understand this fully, it is helpful to further explain these other planning problems along with their inputs and outputs.

The railway planning process at NS is split into six main planning problems: Infrastructure planning, line planning, timetabling, rolling stock planning, crew planning and real-time management. Of these 6, infrastructure planning and real-time management are primarily handled by a secondary firm, while the remaining are mainly performed by NS. These planning problems are highly interdependent, as each planning problem typically requires input from one or more of the other planning problems to be effectively carried out.

In general these planning problems are divided into multiple levels based on their timeline until the implementation of a timetable. Four planning levels are defined: strategic level, tactical level, operational level and real-time level. Strategic level planning at NS is considered as long-term planning, conducted several years to decades before the actual implementation. Tactical level planning follows strategic level planning and spans from one year to several years. Operational level planning involves tasks carried out a few weeks to several months before implementation. Finally, real-time level planning addresses issues as they arise while the timetable is being implemented.

The 6 planning problems can be divided among the four planning levels as seen in Figure 1.1. The arrows in the figure illustrate the chronological order in which these problems are addressed. The planning problems are arranged in this order, because the solution of one problem serves as input for the next planning problem. For example, line planning determines the train lines with the stations that are visited, which is necessary for the execution of timetables to determine the arrival and departure times at the stations.

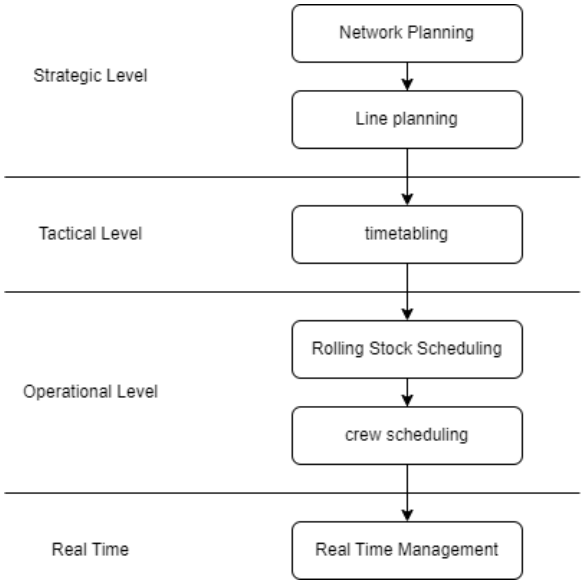


Figure 1.1: Subdivision of the railway planning process by planning levels.

The first step in railway planning is network planning, which involves designing or extending the infrastructural layout of the train network. The output of infrastructure planning consists of two types of details. First, there are the basic structural details of the entire network, including the locations of stations and the rail connections between them. These details are primarily important for strategic level planning, where the focus is on understanding how stations are connected. Second, there are more precise infrastructural details, such as the number and layout of train platforms at stations, the placement of railroad switches, and the number of rails in corridors between stations.. These details are crucial for problems in later planning levels such as timetabling, where safety constraints must be implemented to avoid collisions by ensuring that trains do not run at the same track at the same time.

The second planning problem is line planning, which defines a series of train lines through the previously designed infrastructural network. A train line is a path that provides a direct connection between two stations within the infrastructural network. Each train line includes all the stations where the train service stops, the type of train service and the frequency of train services per hour on that line. NS has three types of train services: Sprinters, Intercity trains, and HSL trains. These types are distinguished by the stations they serve and their travel speeds. Sprinters are used for regional transport and typically stop at every station on their route. Intercity trains are designed for longer distances and usually stop at larger stations, offering good connections between them. HSL trains, on the other hand, use high-speed rail to quickly cover long distances and are used for fast connections between major international stations.

The third planning problem is timetabling and involves creating a timetable for the train lines defined in the line planning problem. These train lines are divided into train services, each running at a frequency of one per hour and a specified direction along the line. Timetabling determines the precise times at which each train service arrives and departs from the stations on its route. Departure and arrival times must be selected to optimize the timetable while adhering to various constraints. These constraints are essential to ensure that the timetable is feasible, safe, and comfortable. Feasibility ensures that there is sufficient time for the train, passengers, and train driver to perform the necessary actions. Safety constraints are needed to maintain sufficient distance between trains to avoid collisions. Comfort constraints are intended to provide passengers with reasonable travel options, although these may be adjusted or ignored as necessary.

The fourth planning problem is rolling stock scheduling, which involves allocating rolling stock to train services. The main factors for this problem include the timetable, the demand for each train connection and the available rolling stock. A critical consideration in this process is cost, as rolling stock represents one of the largest costs in rail transport. The aim is therefore to minimize the costs associated with rolling stock while ensuring that there is sufficient accommodation to meet the demand for each train connection.

The fifth planning problem is crew scheduling, which involves assigning crew members to perform tasks on the train services. This planning problem must comply with several constraints that ensure the rights and well-being of the personnel, such as maximum working hours, break lengths, and other labor regulations. In the Netherlands, crew scheduling starts with creating daily duties, which are collections of multiple tasks. These day duties are then combined into broader crew schedules. This process ensures that constraints such as avoiding repetitive work and achieving a balance of less desirable tasks (referred to as "aggression-work") and more preferred tasks ("A-work") are met.

The final planning problem is real-time management, which addresses problems that arise during the operation of train services. For example, if a train breaks down and can no longer complete its journey, real-time management must quickly reschedule rolling stock and crew to minimize disruption to other train services. While some disruptions can be anticipated and mitigated through resilient planning, real-time management is crucial to quickly respond to unforeseen issues and ensure the smooth operation of the entire train network.

1.2 Thesis contributions and outline

The main topic of this thesis is implementing heuristics for strategic passenger-oriented timetabling (SPOT). This is a variant of timetabling, which is already performed on the earlier strategic level instead of the tactical level with a focus on how travelers experience a given timetable. SPOT is used to form a bridge between line planning, which is mainly passenger-oriented and tactical timetabling, which is focused on producing a feasible timetable for an infrastructural network. SPOT keeps the main point on the passengers such as in line planning, while maintaining some basic restrictions for timetabling. SPOT is focused on providing information to later create a timetable with all infrastructural constraints. Therefore, SPOT is performed earlier than classical timetabling

Chapter 2 starts by explaining the foundation of most timetabling problems, the Periodic Event Scheduling Problem (PESP). PESP is used to create a periodic timetable that repeats every hour. After this, the chapter reviews the literature on strategic timetabling applications and compares them with tactical timetabling. Then, the different focuses of line planning and tactical timetabling in the current literature are reviewed to show how SPOT chooses a middle way between the two by having a passenger-oriented focus for timetabling combined with excluding infrastructural constraints. Finally a description for SPOT is given for which timetables are allowed and how these timetables are evaluated over perceived travel time of passengers. This perceived travel time of passengers is based on the initial waiting time, in-train time, transfer time and penalties for inconveniences related to transfers and initial waiting time.

Chapter 3 discusses how to calculate the perceived travel time from a departure to an arrival for a given timetable by applying Dijkstra's shortest path algorithm over the

train network. This is used to calculate the perceived travel time of passengers for the given timetable. Several options are considered to reduce the calculation time of Dijkstra's algorithm on the train network. After this a simplification for SPOT is introduced, which significantly reduces the problem size by the grouping of passengers, which are known to use the same route. Lastly, Two approximation for SPOT are considered where preselected routes are chosen and a subset of passengers are left out to further reduce the size of the problem.

Chapter 4 explores Polinder's [7] mixed-integer linear programming (MILP) based strategy for solving SPOT, which involves a linear objective function and linear constraints on problem variables, to calculate the optimal timetable. SPOT is first formulated as a non-linear MIP, which is then linearized to create a MILP formulation. This chapter concludes with a discussion of the challenges of this MILP approach, noting that it generally results in an extremely large problem with long calculation times.

Chapter 5 focuses on heuristic approaches for solving SPOT. Two heuristics are considered: Local Search and Simulated Annealing. Both heuristics start with an initial timetable and search for a better timetable in a neighborhood of closely related timetables for improvements. By applying this iteratively good timetables are found. This chapter explains how these closely related timetables are found and how Local Search and Simulated Annealing differ in choosing which timetable is chosen as the new timetable. Lastly, the chapter discusses multiple lower bounds for the perceived travel time of a timetable to assess the heuristics effectiveness and where improvements can be found.

Chapter 6 describes the acquisition of input data for SPOT, considering four types: the train network, passenger demand for connections, penalty variables for the SPOT formulation and the set of preselected routes. The train network data is directly obtained from NS, with some minor modifications. Other data is generated using literature and publicly available data to accurately reflect reality.

Chapter 7 presents the results of choosing different settings for the heuristics. Following this, the optimal settings are used to compare the heuristics against each other. The chapter evaluates how the timetable performs for the passengers. Lastly is considered where the heuristics drastically improve the initial timetable and where it is expected that more improvement can be found.

Finally, Chapter 8 start by extensively summarizing the thesis. After this all conclusions from the found results are noted about how the heuristics and made choices affected the best-found timetables. Lastly, several suggestions are made for potential further research to extend the ideas presented in this thesis.

Chapter 2

Strategic passenger-oriented planning

The focus of this thesis is the strategic passenger-oriented planning (SPOT) , which is a timetabling problem at the strategic level that emphasises the passengers perspective. To begin, it is useful to understand the foundation of most train timetabling problems, which is the periodic event scheduling problem (PESP) developed by Serafini and Ukovich [9]. Following this, the discussion will shift to how strategic timetabling differs from tactical timetabling, their roles in train planning, and the current literature on strategic timetabling. Finally, the exact definition of SPOT will be provided, along with the rationale for adopting a passenger-based perspective, illustrated with an example of solving SPOT.

2.1 Periodic Event Scheduling Problem

The periodic event scheduling problem (PESP) is a widely used formulation for periodic timetabling problems. Initially, an event is defined as the arrival or departure of a train service at a station. Now, periodic event scheduling is defined as:

Definition 2.1. *(Periodic event scheduling) Given a set of events V and a constant T, assign a time $\pi_v \in [0, T)$ for each event $v \in V$. Here, the value T is called the time period of the events and the periodic event schedule involves repeating all events in V every time period T after their original assigned times. Such repeating events are called periodic events.*

In PESP, there are constraints that establish relationships between two periodic events. These constraints set a lower and upper bound on the difference between a periodic event and the next occurrence of another periodic event. These constraints, together with the concept of periodic event scheduling, lead to the following definition by Serafini and Ukovich for the Periodic Event Scheduling Problem (PESP):

Definition 2.2. *(PESP) Given a set of periodic events V and a set of constraints $C \subseteq V \times V$, where a constraint $(i, j) \in C$ consists of a lower bound l_{ij} and an upper bound u_{ij}, find a feasible mapping of $\pi : V \to [0, T)$ such that:*

$$(\pi_j - \pi_i - l_{ij}) \mod T + l_{ij} \in [l_{ij}, u_{ij}], \forall (i, j) \in C.$$

For train timetabling, the constraints can be divided into three basic types: train service defining constraints, safety constraints and comfort constraints. These main types of constraints are described with examples of some of the best-known constraints within each category.

Train service defining constraints: Train service defining constraints form the basis for the feasibility of operating a train service. These constraints are largely dependent on the time required for the crew, passengers, and train service to perform certain tasks. Some examples include:

- **Drive:** Drive constraints are based on operating the direct connection of a train service between two stations without visiting another station in between. For this, the time difference between a departure event and the next arrival event of the train service is restricted by an upper and lower bound. The lower bound for a drive constraint is determined by the distance between the two stations and the speed at which the train travels. The upper bound is typically a slight increase from the lower bound, which can be achieved by adding a small percentage or a fixed value to the lower bound. This upper bound is ideally kept close to the lower bound to ensure that the train service is utilized optimally in the timetable.

- **Dwell:** Dwell constraints are implemented to restrict the time between an arrival event and the subsequent departure event of a train service at the same station. The lower bound for dwell time is based on the time needed for passengers to get on and of the train service, which can be longer at larger stations where more passengers are involved. The upper bound is typically a small increase from the lower bound to ensure optimal usage of the train, similar to drive constraints. However, in the case of dwell times, there can be an incentive to increase the dwell time slightly to facilitate good transfers to other train services.

- **Turnaround:** Most rolling stock units that reach the final destination of their corresponding train service are then used for a train service in the opposite direction. This requires the implementation of turnaround constraints to prepare the train personnel between an arrival event and the subsequent departure event in the opposite direction. The lower bound for this constraint is primarily determined by the time it takes for the train driver to walk to the other end of the rollings stock units before they can start operating the train service in the opposite direction. The upper bound is set to ensure efficient use of the rolling stock units; a longer turnaround time means that additional rolling stock units are needed to maintain the service, leading to higher rolling stock costs.

Safety constraints: Safety constraints are fundamental for a timetable to be operational for multiple train services simultaneously. These constraints are primarily designed to ensure that train services do not get too close to each other while operating on the same track. For this purpose, headway and single track constraints are considered.

- **Headway:** Headway constraints are implemented to prevent collisions between train services traveling in the same direction. This is achieved by ensuring that one train service does not overtake another on the same track between two stations and by enforcing sufficient time between the stops of two train services at the same platform of a station. The lower bounds for these constraints are based on maintaining a safe distance between train services, while the upper bounds directly follow from symmetry: if train service 1 needs enough headway from train service 2, than train service 2 also needs enough headway from train service 1.

- **Single track:** Single track constraints are used when two train services traveling in different directions use the some track. In such cases, there must be a location where these two train services can pass each other. This is typically at stations where the track briefly splits into two. The lower bounds for these constraints are based on the travel time between two points where the train services can safely cross, plus an additional safety margin. Similar to headway constraints, the upper bounds follow by reversing the order in which the train services depart, ensuring that either train service can safely occupy the track.

Comfort constraints: Comfort constraints are introduced to ensure desirable features of a timetable from a passenger perspective. While these constraints are not necessary for the timetable's implementation, they significantly enhance the passenger experience. Two different comfort constraints are transfer constraints and frequency constraints.

- **Transfer:** Transfer constraints are used to improve connections between the arrival event of one train service and the departure event of another train service at the same station. Not every transfer is constrained, as there is no incentive to improve transfers for train services with the same travel path or those traveling in opposite directions. Additionally, adding too many low upper bounds for transfer constraints can make the problem infeasible. The lower bound of a transfer constraint is based on the minimum time required for passengers to leave one train service, change platforms, and board the next train service. The upper bound is chosen to ensure that the transfer time is not excessively long.

- **Frequency:** Frequency constraints are used to ensure that multiple train services traveling in the same direction are evenly distributed over the time period. For example, if there are four train services per hour, the train services are scheduled so that a train service operates approximately every 15 minutes. This is achieved by setting a lower bound slightly less than 15 minutes and an upper bound slightly more than 15 minutes, ensuring that the trains are spread out evenly throughout the hour.

2.1.1 Visualization of PESP example.

To visualize PESP, we will refer to an example from Polinder [7], which visualizes PESP. In this example, the line planning of three train lines, each with a frequency of one per

hour and their drive constraints, is illustrated in Figure 2.1. For this scenario, a time period of one hour is used. the solid train line represents an intercity train that visits S_1, S_2, S_3, S_4 but does not stop at S_2. The dashed train line represents a sprinter that visits S_1, S_2, S_3, S_4, stopping at all stations. The dotted train line represents another sprinter that visits S_5, S_3, S_6, also stopping at all stations.

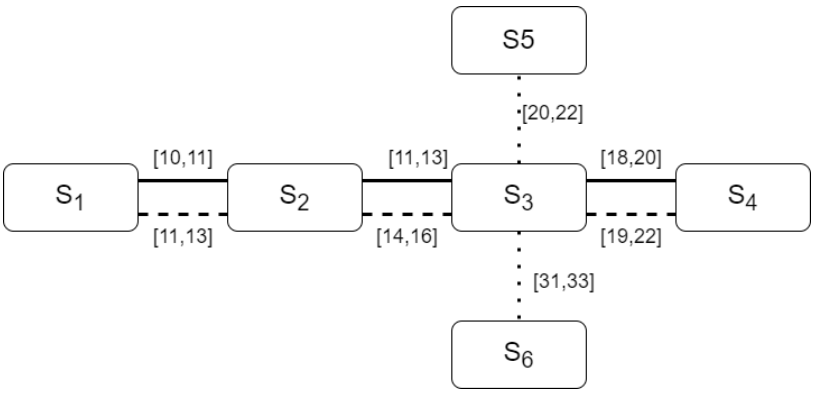


Figure 2.1: Schematic view of three train lines (solid, dashed and dotted lines) combined with the constraints for their drive times from station to station.

This line plan is extended to include all arrival and departure events of the train lines, with all missing constraints added. This represents the event-constraint network shown in Figure 2.2. In this network, each grey circle represents a departure event and each white circle represents an arrival event. The solid, dashed and dotted train lines of the line planning are now referred to as L_1, L_2 and L_3 respectively. On the left of every row of events the train line is given to which all events belong to. For every train line there are two train services, one for the forward direction and the other for the backward direction. In the middle of each grey box, the station is indicated to which all events in the box belong. Every connection from a departure to an arrival contains the drive constraint for the train service. A connection on the same row from an arrival to a departure has a dwell constraint, which is between 1 and 3 minutes if the train stops and 0 minutes if the train does not stop, as seen for train line L_1 at station S_2. The lines on the side represent turnaround constraints for taking a train service in opposite direction, which is between 7 and 56 minutes. The constraints between two arrivals or departures are headway constraints, ensuring that two trains services in the same direction always need 3 to 57 minutes between them. Finally, transfer constraints are used to enforce good transfers at station S_3.

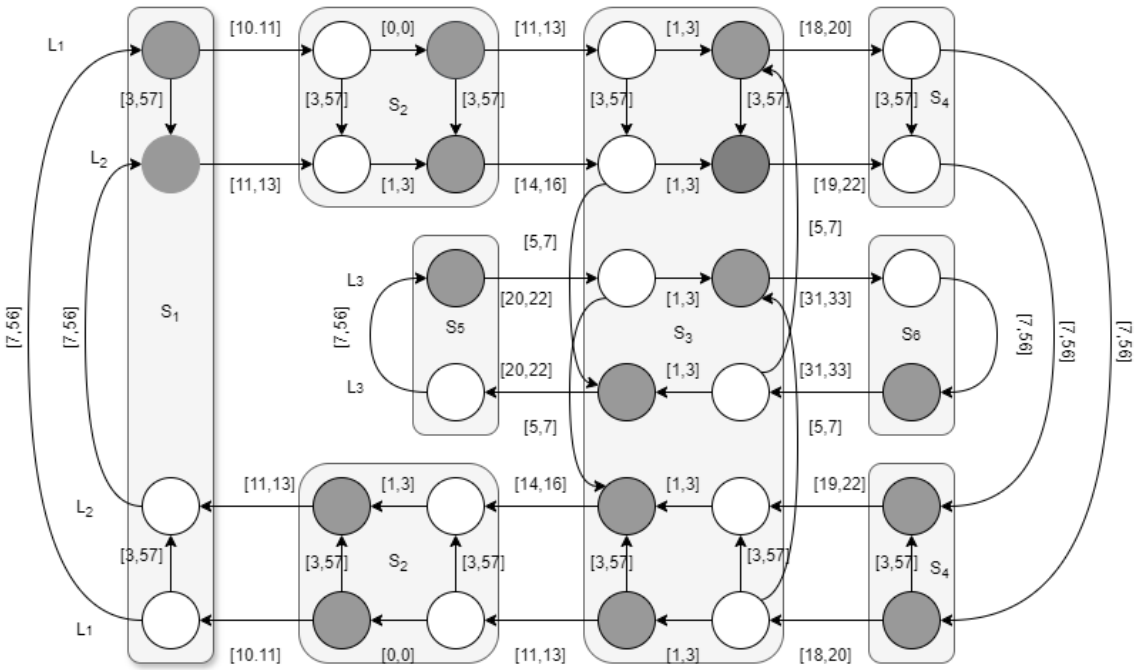


Figure 2.2: Example of an event constraint network for PESP.

2.2 Strategic timetabling

In this thesis, we focus on strategic timetabling, which is conducted multiple years to decades before the implementation of a train schedule. While timetabling is typically defined as a tactical level problem spanning a time horizon of one to several years before implementation. The primary goal of strategic timetabling is not to create timetables for implementation, but to address strategic questions for later planning problems. For example, strategic timetabling helps determine which transfers are crucial for an effective timetable. Since strategic timetabling is performed so early, it allows for the removal of infrastructural constraints, enabling a more passenger-oriented approach. This results in the creation of timetables that provide good connections for passengers, which are used in subsequent planning stages to develop timetables that incorporate infrastructural constraints. In addition to gathering information for future planning problems, strategic timetabling can reflect on other strategic level planning problems. For instance, it can assess the effectiveness of a line plan and identify areas where it does not perform well. Furthermore, strategic timetabling helps to identify bottlenecks in the current infrastructure, guiding decisions on investments needed to address these bottlenecks and improve the overall network.

In recent years, the use of strategic timetabling has been increasingly evident in the evaluation of infrastructural networks. For example, Sartor et al. [8] researched how differ-

ent infrastructural improvements resulted in better timetables for the Jaeren line in Norway. Their findings were utilized by the Norwegian Railway Directorate to select infrastructural enhancements for the expansion of the line. As another example, currently in Germany there is a plan called "Deutschland Takt" which aims to make local and long-distance travel by train more reliable. Germany wants to achieve what already works in Switzerland, where cyclic timetables are used to consistently drive connections between stations. In this way, Germany tries to create a target timetable where every train line is operated every hour or every 30 minutes for long distance trains. Based on this target timetable, infrastructural bottlenecks are identified and prioritized for improvement in the German train network.

There are multiple papers about integrating timetabling into the line planning problem. For example, Burggraeve et al. [2] and Yan and Goverde [11] proposed iterative methods to produce effective line plans. In these approaches, an initial line plan is created and evaluated using various factors, such as travel time and the number of lines. Based on the insights gained from timetabling, critical issues in the line plan can be identified. These shortcomings are then addressed by creating a new line plan, and this process is repeated until the line plan and timetable meet certain criteria. In addition Zhang et al. [12] proposed an integrated optimization of line planning and train timetabling using a mixed-integer linear programming (MILP) model.

2.3 Problem definition

For this thesis, a specific type of strategic timetabling called Strategic Passenger-Oriented Timetabling (SPOT) is used, as proposed by Polinder [7]. As indicated by the name, this timetabling problem focuses on the perspective of passengers. This approach is adopted due to the difference in focus between line planning and tactical timetabling.

Line planning tends to emphasize passenger perspectives, as seen in early works such as Bussieck [3], which focused on maximizing the percentage of travelers reaching their destination without transferring to another train. Later research, such as Bull et al. [1], includes considerations for indirect travelers by accounting for expected travel time. For line planning, only basic features of the infrastructural network need to be known, such as how stations are connected to each other so that a train service can be performed between two stations.

Tactical timetabling in academic research primarily concentrates on the feasibility of a timetable for an infrastructural network. This feasibility-related timetabling is often restricted due to infrastructural constraints, resulting in reduced quality for passengers. Furthermore, passenger quality is based on comfort constraints, which are derived from the expected structure of a timetable that is assumed to be good for passengers, but this assumption cannot always be confirmed. In Harrod [6] and Caimi et al. [4] an extensive list of timetabling problems are given

SPOT bridges the gap between line planning and tactical timetabling by creating timetables with less focus on infrastructural feasibility and more focus on the passenger perspective. By doing so, timetables that are beneficial for passengers are developed, from which either useful information can be acquired or adjustments can be made to ensure they work within the infrastructural network, as introduced by Polinder [7].

2.3.1 Passenger demand and perceived travel time

The first key concept in SPOT is passenger demand, which is determined by the start station, end station, and preferred departure time of passengers. Passenger demand is defined as the number of passengers per origin-destination-time triple (ODT-triple), with time represented in intervals of equal size due to the impracticality of using exact preferred departure times. for this the preferred departure time of an interval is set as the average of the interval. Since SPOT uses periodic time, the demand for an ODT-triples is calculated as the average ODT demand over multiple periods. Data for each time interval is challenging to acquire, therefore is chosen for uniformly spreading the amount of passengers per origin-destination pair (OD-pair) over the ODT-triples. Now, d_k is defined as the number of passengers for OD-pair $k \in OD$ per time period. Then given the set I containing all time intervals I_t with preferred departure time t, there are $\frac{d_k}{|I|}$ passengers per time interval for OD-pair k.

For SPOT, the evaluation of a timeatble is based on the perceived travel times of the passengers of ODT-triples. Here, a route r consists of an ordered sequence of subsequent drive, dwell and transfer activities. The activity durations of these drive, dwell and transfer activities are bounded by their corresponding drive, dwell and transfer constraints. Now, the perceived travel time for passengers in a ODT-triple given a route is a variation on the actual travel time, where is considered that certain tasks are inconvenient for passengers. In the following two paragraphs, the perceived travel time over a route and the perceived waiting time before the first departure of the route are explained. These two together will define the perceived travel time of an ODT-triple for a given route.

The perceived travel time over a route is a variation on the real travel time of a route, which is defined by the sum of all times of the activities in the route. instead, the perceived travel time over a route also includes the inconvenience experienced by a passenger during transfers. This inconvenience arises because a passenger needs to walk to the next train and there is always some uncertainty about whether the next train will be on time. To account for this inconvenience, a transfer penalty γ_t is introduced for every transfer on the route. Thus, the perceived travel time Y_r of a route r subject to timetable π with activity durations y_{ij} is defined as:

$$Y_r = \sum_{(i,j)\in r} y_{ij} + \gamma_t \cdot 1_t(i,j).$$

Here $1_t(i,j)$ is an indicator function, which is equal to 1 if $(i,j) \in A$ is a transfer activity and 0 otherwise.

Apart from the perceived travel time of a route, SPOT also includes the initial waiting time at a station before the first departure of the route. It can be argued that passengers prefer an earlier departure with a slightly longer route over a later departure that arrives earlier, as this provides certainty that the train service will drive as planned. Because of this, a penalty γ_w is introduced for the initial time waited before the first departure of a route. Let W_r^t be the initial waiting time of a passenger with preferred departure time t and taking route r, with initial departure event $\sigma(r)$ for a given timetable π. Then W_r^t is defined as:

$$W_r^t = \pi_{\sigma(r)} - t \mod T$$

The perceived travel time Y_r^t of a passenger with preferred departure time t with route r is then defined as:

$$Y_r^t = \gamma_w \cdot W_r^t + Y_r$$

2.3.2 Route choice

Given the passengers in an ODT-triple, there are typically multiple routes available for traveling between the start and end station. It can even be noted that there are an infinite amount of routes between two stations as it is possible to go to another station and return back to extend a route. Therefore, a decision must be made regarding which route the passengers will take among the available options. In SPOT, it is assumed that each passenger will choose the route that minimizes their perceived travel time.

2.3.3 Objective value

Based on all earlier mentioned definitions of passenger demand, perceived travel time and route choice, an objective value is defined that minimize the perceived travel time of a timetable π. For this the perceived travel time of a timetable is defined as the average perceived travel time of the passengers:

$$tt = \frac{1}{D} \sum_{k \in OD} \sum_{I_t \in I} \frac{d_k}{|I|} \cdot \min_{r \in R^k} Y_r^t. \tag{2.1}$$

Here we sum over all minimal perceived travel times of origin destination pairs $k \in OD$ for all given time intervals I_t with passenger demand $\frac{d_k}{|I|}$. D is here the total sum over all passenger demands to create the average perceived travel time of a passenger and R^k is the set of routes for OD-pair k.

2.3.4 SPOT constraints

The foundation of timetables in SPOT lies in cyclic timetabling based on the Periodic Event Scheduling Problem (PESP). However, SPOT places much less emphasis on the feasibility of a timetable compared to the original form of tactical timetabling. Instead of aiming for an operable timetable, SPOT focuses on minimizing an objective value based on the travel

time of passengers. As a result, many constraints that are typically included in PESP for tactical timetabling are disregarded.

Comfort constraints are used to ensure preferential characteristics from a passenger's perspective in a timetable. However, since SPOT is designed to optimize timetables with a focus on passenger experience, it inherently addresses many of these comfort aspects. Therefore, comfort constraints are excluded in SPOT. Despite this, transfer times are crucial for SPOT, therefore transfer constraints are designed which do not exclude any timetable. Specifically, the lower bound l_{ij} for the transfer from arrival event i to departure event j is the minimum time required to exit the train service, travel to the next platform, and board the subsequent train service. The upper bound u_{ij} is set to the lowest value such that no time table is excluded, resulting in $T + l_{ij}$.

Safety constraints are for Strategic timetabling not strict constraints due to the uncertainty about the exact infrastructure that is used by a train service. Also, many headway constraints will still hold automatically since a good spreading of the trains is often preferred by passengers. Because of these two reasons, all safety constraints will be ignored for SPOT. This makes it easier to create feasible timetables and thus also easier to search for good feasible timetables.

Most of the train line defining constraints remain essential for SPOT because they ensure that train services are operational. These constraints are crucial for maintaining the feasibility of a timetable, as they dictate the basic operational requirements, such as travel times between stations and required dwell times for passengers. However, turnaround constraints are excluded in SPOT. This is because it is not necessary for the same rolling stock units to be used for a train service in the opposite direction.

The removal of safety and comfort constraints eliminates the restrictions between different train lines, and the removal of turnaround constraints removes the restrictions between different train services within the same train line. As a result, the event-constraint network is now only constrained by events of the same train service, which is illustrated in Figure 2.3. Notably, train line L_1 no longer includes the arrival and departure events at station S_2, which previously represented the train passing through the station. These events were initially included to account for safety constraints related to passing through the station, but since safety constraints have been removed, these events are no longer necessary. This simplified PESP example is relatively straightforward to solve, as the times for the train services can be calculated separately. This means that the real challenge in PESP came from the constraints that have been removed.

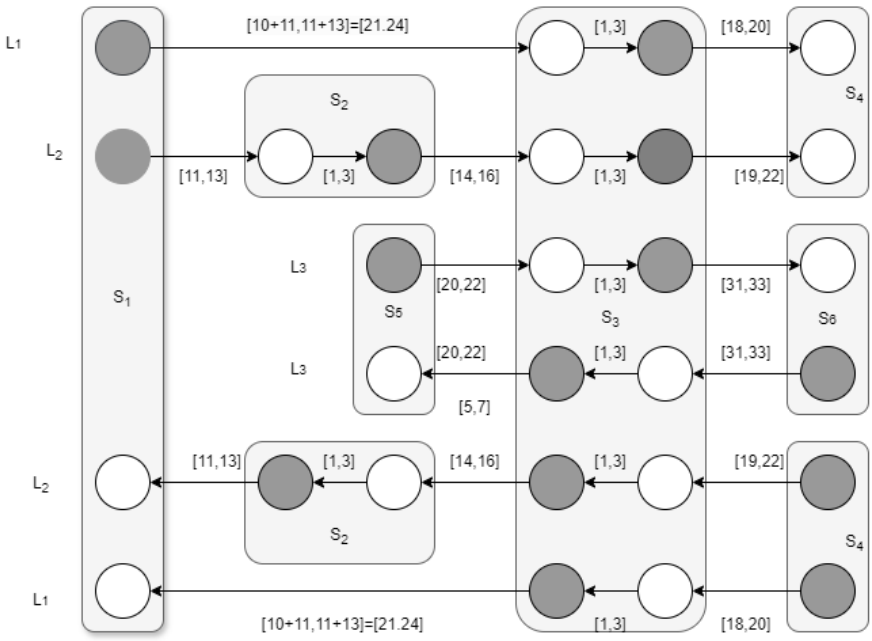


Figure 2.3: Representation of an example of PESP for SPOT.

2.4 Example of SPOT

Consider a subset of the original network shown in Figure 2.3 by removing train line L_3 and station S_4. The new event-constraint network, as depicted in Figure 2.4, includes all remaining drive, dwell and transfer activities. For the drive and dwell activities, the bounds are specified, while the duration of a transfer activities are assigned a bound of $[3, 62]$. In this example, there are 3 potential start stations, all with 2 potential end stations, which results in 6 OD-pairs. The passenger demand for all OD-pairs will be set to 60 passengers every hour and thus 1 per minute. Initially, let $\gamma_t = M$ and $\gamma_w = 1$, where M is a very large value intended to force direct routes.

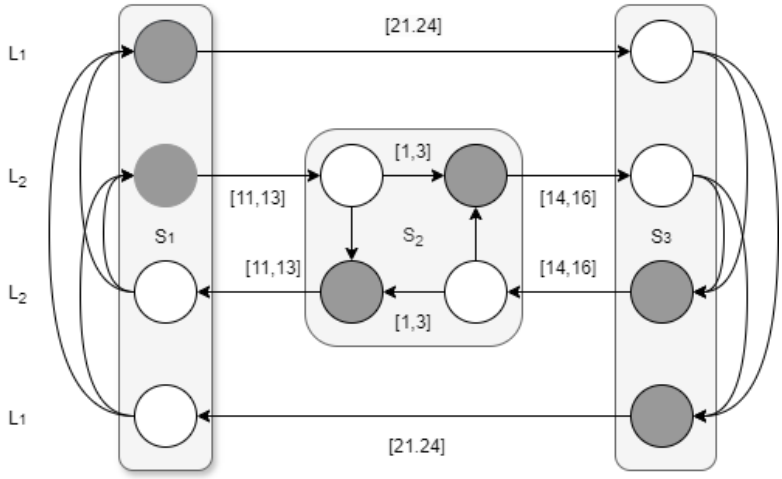


Figure 2.4: Example constraint-event network of an SPOT example.

Since transfers are heavily penalized, the problem effectively splits into two identical problems: one for the train services from S_1 to S_3 and another for the train services from S_3 to S_1. In this example most passengers have few choices, passengers between S_1 and S_2 have one train per hour, taking 11 minutes and passengers between S_2 and S_3 have one train per hour, taking 14 minutes. Only passengers between S_1 and S_3 have two possible routes, one taking 21 minute via L_1 and another taking $11 + 1 + 14 = 26$ minutes via L_2. The timetable is optimal if L_2 departs 32 or 33 minutes after L_1 from the start station. With this an optimal solution can be seen in Figure 2.5

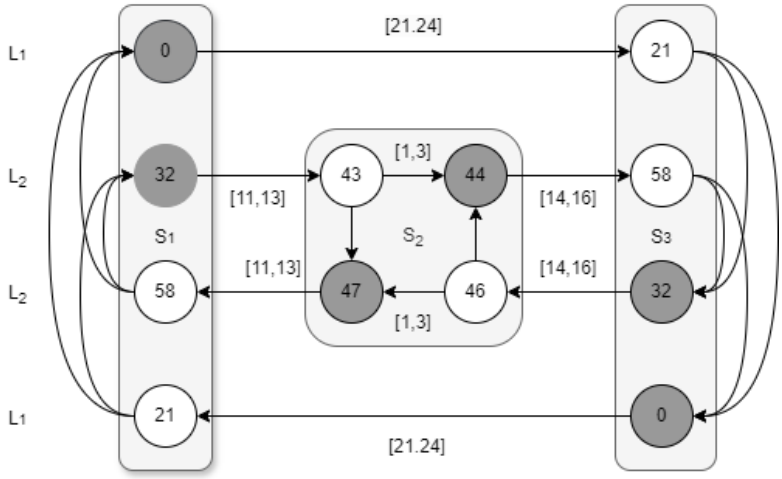


Figure 2.5: Optimal timetable for a SPOT example with large transfer penalty.

Consider that $\gamma_t = 5$, introducing a lower transfer penalty. This adjustment results in multiple possible routes from or to station S_2 that, while not direct, are still faster. For example, a route from S_1 to S_2 could take L_1 to go from S_1 to S_3 and then L_2 to go from S_3 to S_2. This results in a better timetable shown in Figure 2.6. In this scenario, the new route from S_1 to S_3 takes 43 perceived travel minutes, consisting of 38 real travel minutes and 5 transfer penalty minutes. If a passenger wants to travel from S_1 to S_3 with preferred departure time 0, taking this new route is faster than waiting for the direct train over L_2, which takes 44 perceived travel minutes.

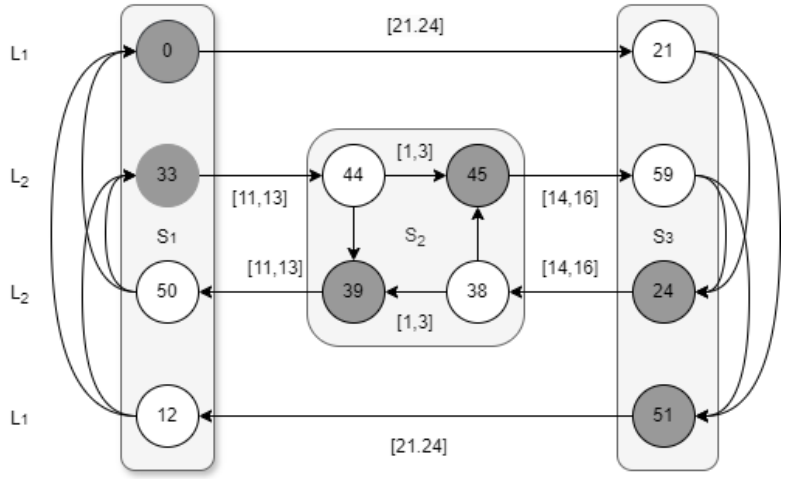


Figure 2.6: Optimal timetable for a SPOT example wit a transfer penalty of 5 minutes.

Apart from the the transfer penalty, the initial waiting penalty γ_w and the OD-matrix also significantly influence an optimal timetable. If γ_w were larger, it would force a more equal distribution of waiting times for passengers taking L_1 and L_2 from S_1 to S_2, leading to departures closer to 30 minutes apart. In this specific example, another OD-matrix has no influence. However, in scenarios where a few OD-pairs are much more critical than others, some connections might be worsened to improve the more important ones.

Chapter 3

Calculating perceived travel time and problem reductions

This chapter focuses on how to calculate the objective value of a given timetable and how to simplify and approximate the problem. First, Dijkstra's shortest path algorithm is described, which calculates the route with the shortest perceived travel time from a departure to another station. Several changes are made to reduce the calculation time for Dijkstra's algorithm. A simplification for SPOT is described, where passengers are grouped together if it is certain that they will take the same route. Finally, two approximations for SPOT are given based on preselecting routes and reducing the OD-matrix.

3.1 Calculating the shortest route

It can be difficult to calculate a passenger's perceived travel time for a given timetable because the set of possible shortest routes for a passenger can become large. Of course, there are routes a passenger never takes, but even after these are removed, many remain. By the additive structure of the perceived travel time over the activities, it is possible to calculate the perceived travel time step by step by adding a drive, dwell or transfer activity to the route. Instead of checking all routes between two stations, the additive structure allows to use Dijkstra's shortest path algorithm, which is used for calculating the shortest route between a departure and arrival event without a set of possible routes.

3.1.1 Dijkstra's shortest path algorithm

Dijkstra's shortest path algorithm is used to find the shortest path between vertices in a weighted graph with non-negative weights. For SPOT this weighted graph given a timetable π is a copy of the event-constraint network were the events are the vertices and the constraints are changed into the perceived travel time of performing the constrained activity.

As an example, the weighted graph of the event-constraint network shown in Figure 2.6 is shown in Figure 3.1. Here the constraints are changed into the perceived travel times of

the activities. For drive and dwell activities, this is the real time that the activity takes to perform and in the case of transfer activities, it is the real time plus the transfer penalty γ_t. In this graph Dijkstra's algorithm needs to find the shortest route from a departure event to arrival events.

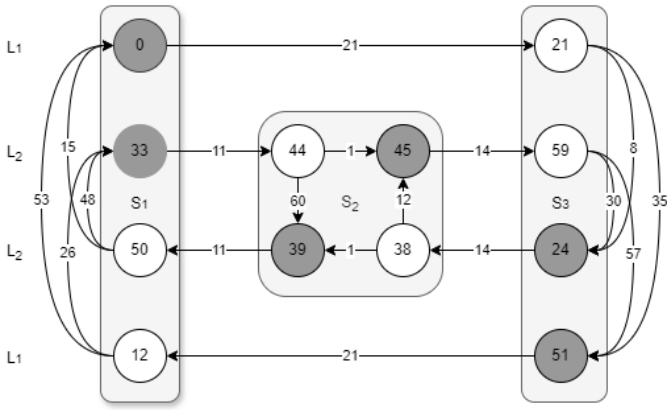


Figure 3.1: Activity-event network for Dijkstra's algorithm.

In basic form, Dijkstra's algorithm calculates the perceived travel time from an event to all other events. First a set of unvisited events is created containing all events for which the shortest route from the starting event has not yet been found. Then each event is given a distance value, which is 0 for the starting event and infinity for all other events. This value represents the currently shortest route from the start event to the event, with infinity indicating that a route to the event has not yet been found. Now the following steps are performed until a shortest route to each event is found.

1. Find the event with the shortest distance value, which is still in the set of unvisited events. This event is the starting event in the first iteration. In case no event occurs in the set of invested events, the algorithm ends with step 4.

2. The found event is called the current event, over which a route to all neighbours in the set of unvisited events can be calculated. This route through the current event has a distance value equal to the distance value of the current event plus the weight from the current event to the neighbour. The distance value of the neighbor is then updated to the smaller value between the distance over the current route and the previous distance value of the neighbor.

3. The current event is removed from the set of unvisted events and its distance value will no longer be updated. This is because, given step 1 and non-negative weights, all routes through other events cannot be shorter than the current distance value. After this, step 1 is repeated.

4. Now, the final distance value to each event from the starting event has been determined, representing the shortest route from the starting event to each respective event. The perceived travel time can be infinity if no route existed.

Dijkstra's algorithm runs in $O(|E|+|V|log(|V|))$ by implementing a Fibonncci-heap for the distance values, which makes that searching for the minimum distance value is done in $log(V)$.

Instead of running Dijkstra's algorithm to compute the perceived travel time from a departure event to all other events, it is possible to determine the perceived travel time to a subset of the other events by stopping when all these events are visited. For SPOT, the perceived travel time from an departure to the other stations is needed. The perceived travel time to a station is determined by the lowest perceived travel time to an arrival event at that station. Specifically, the arrival event at a station that is visited first has the same perceived travel time as that of the station itself. Thus, the iterative steps can stop when at least one arrival event of every station has been visited.

A single run of Dijkstra's algorithm for the weighted graph in Figure 3.1 is shown in Figure 3.2. Starting from the departure event of train service L_1 at S_1, The green event represents the current event, the red events denote those already visited and the blue stations indicate those still to be visited. In the first iteration, the start event becomes the current event, which has as neighbour the arrival at S_3. This arrival at station S_3 receives a distance value of $0 + 21 = 21$ minutes. In the next iteration, the arrival at S_3 is the closest unvisited event, and as the first visited arrival at S_3, the perceived travel time from the departure event of train service L_1 at S_1 to S_3 is 21 minutes. The arrival event at S_3 has two neighbouring departure events at S_3, leading to distances of $21 + 8 = 29$ minutes and $21 + 35 = 56$ minutes. In the subsequent iteration, the departure of L_2 at S_3 is the closest unvisited event, resulting in a distance value of $29 + 14 = 43$ minutes to the arrival event at station S_2. In the final iteration, the arrival at S_2 is the closest unvisited event, making it first visited arrival at S_2 and giving a perceived travel time to S_2 is 43 minutes. Therefore the final perceived travel times from the departure event of L_1 at station S_1 is 21 minutes to S_3 and 43 minutes to S_2.

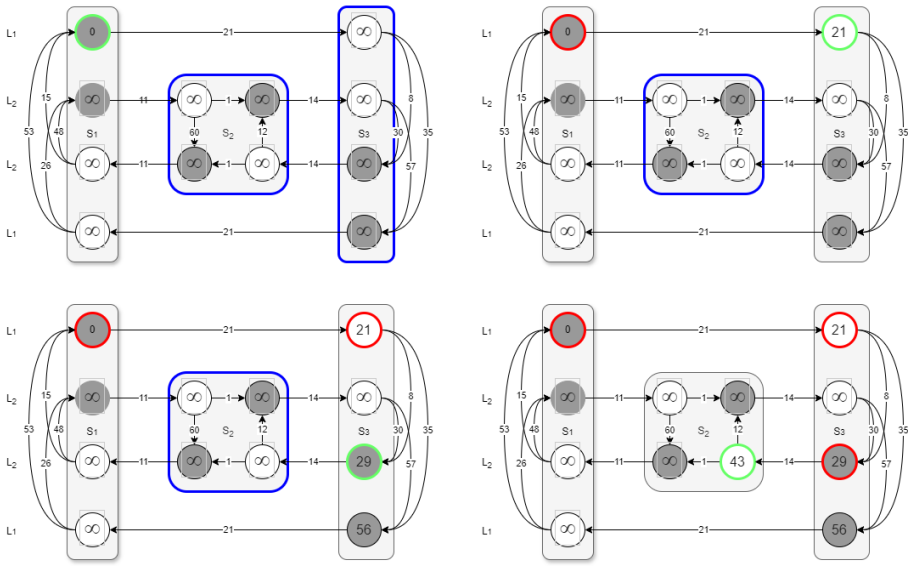


Figure 3.2: representation of an example PESP problem

By running Dijkstra's algorithm for each departure event from a station to all other stations, the shortest perceived travel time for a passenger is determined. This involves minimizing the sum of the waiting time for a departure and the perceived travel time from that departure to the final destination. Consequently, Dijkstra's algorithm is executed for every departure from every station to compute the shortest perceived travel times to all other stations.

3.1.2 Improvements on the graph representation for Dijkstra's algorithm

Dijkstra's algorithm is slower the more activities, for which transfer activities are a large part. To manage the high number of transfers, waiting chambers are implemented at each station. These waiting chambers are designed to connect arrivals to departures via events that account for waiting at the station until the time of the next departure. Each departure event is connected to a waiting event that has the same time as the departure, resulting in a perceived travel time of 0. Additionally, each waiting event is linked to the subsequent waiting event for the next departure from the station. Arrivals at a station must still be directly connected to the departure of the same train service to account for dwell time. Instead of connecting an arrival directly to all other departures, it is connected to the first waiting chamber that occurs after the minimum transfer time. This connection represents that a transfer happens and is assigned a weight equal to the transfer penalty plus the time difference.

In Figure 3.3, an example illustrates how these waiting chambers modify the event-activity network at a single station for 4 train services, considering a transfer penalty of 5

minutes. On the left side of the figure, the network is shown without waiting chambers, while the right side includes them. The white events represent the arrival events and the grey events denote the departure events along with their corresponding waiting chambers. The values within the events indicate the timetable times for these events.

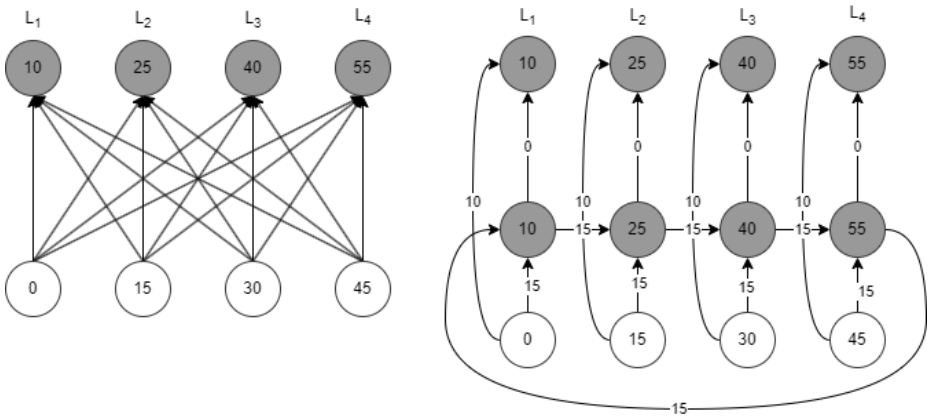


Figure 3.3: Event-activity networks of 4 train services at one station for (left) without waiting chambers (right) with waiting chambers.

The original and new event-activity network are functionally identical for Dijkstra's algorithm. A route from an arrival to a departure at a station in the original network is also present in the new network with the same perceived travel time. If the arrival and departure use the same train service, the original edge remains unchanged. For a transfer to another train service, the transfer penalty is directly included when waiting until the earliest possible waiting chamber that meets the minimum transfer time requirement. All other waiting times are accounted for by traversing through the subsequent waiting events until reaching the waiting event for the departure, which is connected with the departure with a weight of 0.

Instead of finding the shortest path from a departure at a station to the closest arrival at another station, an alternative approach is to reverse the direction of all edges and run the algorithm from the arrivals back to the initial departures. In this method, all arrival events at a station are initially assigned a distance value of 0, while all other events are set to a distance value of infinity. By running Dijkstra's algorithm from these arrivals to the departures, the shortest route from the departures to the station are determined. This approach allows to calculate the perceived travel time from all departures to a station, by applying Dijkstra's algorithm only ones, rather than using it for each departure. This reduces the computational effort significantly by limiting the number of runs of Dijkstra's algorithm.

3.2 Grouping of passengers

In the objective function (2.1) we have a summation over every OD-pair $k \in OD$ for each time interval of passengers I_t. For each of these $|I| \cdot |OD|$ scenarios, an optimal route must be selected. While routes for different OD pairs cannot be the same, there is a possibility that the shortest route for the same OD-pair are the same across different time intervals.

Consider two passengers with different preferred departure times, A and B, who have the same OD-pair, with A having an earlier preferred departure time than B. If there is no train departure from the start station between the preferred departure time of A and B, then the travel route for A is the same as the travel route for B. Consequently, all travelers with a preferred departure time between a train departure and the next train departure in any direction from the same station will have the same optimal route. Given a timetable π, this implies that passengers for an OD-pair $k \in OD$ can be divided into time slices S_v^k based on the first departure event $v \in V^k$. Thus the perceived travel time Y_v^k from departure event v for OD-pair $k \in OD$ can be defined as

$$Y_v^k = \min_{r \in R^k} \gamma_w \cdot (\pi_{\sigma(r)} - \pi_v \mod T) + Y_r$$

Now the perceived travel time Y_v^k represents the perceived travel time of all passengers in time slice S_v^k after waiting until departure v.

For a time slice S_v^k with a length of L_v^k, the passenger demand and average waiting time in the time slice needs to be defined. Since passengers are uniformly spread over the time intervals follows that the passenger demand for the time slice is $\frac{d_k}{T} \cdot L_v^k$ and the average waiting time in the time slice is $\frac{1}{2}L_v^k$.

By combining the waiting time until an event v and the perceived travel time from event v, the total perceived travel time for passengers in time slice S_v^k is:

$$\frac{d_k}{T} \cdot L_v^k(\gamma_w \cdot \frac{1}{2}L_v^k + Y_v^k)$$

from this, the objective function for the average perceived travel time per passenger is:

$$tt = \frac{1}{D} \sum_{k \in OD} \frac{d_k}{T} \sum_{v \in V^k} L_v^k(\gamma_w \cdot \frac{1}{2}L_v^k + Y_v^k)$$

.

3.3 Approximations for SPOT

3.3.1 Preselected routes

The set of routes for OD-pair $k \in OD$ has been considered as the set of all possible routes. However, many possible routes have a much longer estimated route than other routes for

the same OD-pair. In addition, routes with many transfers are almost never taken by passengers. By removing routes, which are unlikely to be used, a set of preselected routes is created. When this set of preselected routes is small, the objective value can be evaluated quicker by calculating the perceived travel time over all these routes, rather than applying Dijkstra's algorithm to the entire network. This preselected set of routes is determined prior to the development of any specific timetable and is based solely on known bounds for drive, dwell, and transfer times. Here, transfer times are mostly dependent on the frequency of direct trains between stations.

Deciding if a route needs to be included can be done by evaluating routes based on their minimal, expected and maximal perceived travel times. The expected travel time is determined by using the shortest dwell and drive times and estimating transfer times based on the number of possible transfers in the same direction as the route. The initial waiting must also be considered in both the maximal and expected perceived travel times, as it is an significant part of the objective value.

The most basic tactic ensures that a route is included if there is any chance that a passenger might use the route. For this, every route for an OD-pair is accepted if its minimal travel time is shorter than the lowest maximum travel time of all routes for that OD-pair. Alternatively, a smaller set of routes can be selected by evaluating the correlation between the minimal and expected travel times for routes within the same OD-pair. This tactic provides an approximation for the objective value, as some routes that passengers might chose are excluded. Consequently, the preselected set of routes creates an upper bound for the objective value, because some passengers may not be able to take the shortest possible route.

The use of preselected routes also affects the grouping of travelers. Instead of creating a time slice for every departure event from a station, only the departure events v for which there exists a route $r \in R^k$ such that $\sigma(r) = v$ need to be considered. This limits the time slices to those corresponding to the possible departures of the preselected routes of an OD-pair.

3.3.2 Reduction of the OD-matrix

Until now, all OD-pairs have been considered for calculating the average perceived travel time of all passengers. However in real-data there are few OD-pairs with very large passenger demands and many OD-pairs with very small passenger demands. Therefore, a large part of the OD-pairs can be removed, while keeping a large part of the passengers. difference between.

The OD-reduction is performed by removing the smallest OD-pairs until either a certain percentage of the OD-pairs or total passengers is kept. If the amount of removed OD-pairs is not too high, then the remaining OD-pairs in the reduced OD-matrix force good connections for the removed OD-pairs.

The smaller removed OD-pairs are often for small stations far away from each other. These OD-pairs often have multiple transfers and many possible routes. This makes that the percentage of removed routes from the removed OD-pairs is higher than the percentage of removed OD-pairs resulting in a significant decrease of the amount of routes that need to be checked. The longer routes for the removed OD-pairs generally means that the perceived travel time for reduced OD-matrix is lower than the complete OD-matrix.

Chapter 4

Polinder's MILP formulation for SPOT

This chapter elaborates on Polinder's method for solving SPOT using a Mixed-Integer Linear Programming (MILP) formulation. The model is first presented in a Mixed-Integer Programming (MIP) format, with each constraint explained step by step. Following the MIP formulation is linearized by introducing new variables. Finally, the complete MILP formulation will be presented, and the challenges associated with the model are discussed.

4.1 Mixed Integer Programming Formulation of SPOT

$$\min \quad \frac{1}{D} \sum_{k \in OD} \frac{d_k}{|I|} \sum_{v \in V^k} L_v^k \cdot (\gamma_w \cdot \frac{1}{2} L_v^k + Y_v^k) \tag{4.1}$$

$$\text{s.t.} \quad y_{ij} = \pi_j - \pi_i + T \cdot p_i \qquad \forall (i,j) \in A \tag{4.2}$$

$$l_{ij} \leq y_{ij} \leq u_{ij} \qquad \forall (i,j) \in A \tag{4.3}$$

$$Y_r = \sum_{i,j \in r} (y_{ij} + \gamma_t \cdot 1_t(i,j)) \qquad \forall r \in R \tag{4.4}$$

$$\Delta_{v,v'} = \pi_{v'} - \pi_v + T \cdot a_{v,v'} \qquad \forall v, v' \in v^k \tag{4.5}$$

$$L_v^k = \min_{v' \in V^k \setminus \{v\}} \Delta_{v',v} \qquad \forall k \in OD, v \in V^k \tag{4.6}$$

$$a_{v,v'} + a_{v',v} = 1 \qquad \forall k \in OD, v, v' \in V^k, v \neq v' \tag{4.7}$$

$$Y_v^k = \min_{v' \in V^k} \min_{r \in R_{v'}^k} Y_r + \gamma_w \cdot \Delta_{v,v'} \qquad \forall k \in OD, v \in V^k \tag{4.8}$$

$$L_v^k \in [0, T] \qquad \forall k \in OD, v \in V^k \tag{4.9}$$

$$Y_r \in \mathbb{R}^+ \qquad \forall r \in R \tag{4.10}$$

$$Y_v^k \in \mathbb{R}^+ \qquad \forall k \in OD, v \in V^k \tag{4.11}$$

$$\pi_v \in \{0, 1, \ldots, T-1\} \qquad \forall v \in V \tag{4.12}$$

$$p_{ij} \in Z^+ \qquad \forall (i,j) \in A \tag{4.13}$$

$$a_{v,v'} \in \{0, 1\} \qquad \forall k \in OD, v, v' \in V^k, v \neq v' \tag{4.14}$$

Here, (4.2) and (4.3) represent the PESP constraints. Constraint (4.4) gives the perceived travel time Y_r for a route r by summing over the perceived time of all activities in r. Constraints (4.5), (4.6) and (4.7) are newly introduced constraints by Polinder et al. [7], which calculate the lengths of the time slices. For time slice S_v^k with departure event v and OD-pair k, the first departure event $v' \in V^k$ that happens before departure event v needs to be identified. Constraint (4.7) determines the order of the departure events in v^k, even when two departures occur simultaneously. Constraint (4.5) calculates the time between all departures $v' \in V^k$ to v and constraint (4.6) finds the shortest time, which is the time between v' and v. Constraint (4.8) minimizes the route for time slice S_v^k starting at departure v over all possible departure events in V^k and their possible routes. All other constraints specify the allowed values for the variables.

4.2 Linearization of SPOT formulation

In the MIP there are three parts that are not linear. The first is the objective function (4.1), which contains the quadratic term $L_v^k \cdot (\gamma_w \cdot \frac{1}{2} L_v^k + Y_v^k)$. The second is the minimization in constraint (4.6) for obtaining the length L_v^k of time slice S_v^k. The third is the minimization in constraint (4.8) for determining the perceived travel time Y_v^k for time slice S_v^k.

4.2.1 Objective function

Polinder et al. [7] linearized the objective function by introducing the new variables $x_{v,d}^k$. First, the objective function is rewritten as follows:

$$\min \quad \frac{1}{D} \sum_{k \in OD} \frac{d_k}{T} \sum_{v \in V^k} \gamma_w \cdot \frac{1}{2} (L_v^k)^2 + L_v^k \cdot Y_v^k$$

and the new variables are defined by:

$$\begin{aligned} L_v^k &\geq d \cdot x_{v,d}^k \\ L_v^k &\leq d \cdot (1 - x_{v,d}^k) - 1. \end{aligned}$$

The values $x_{v,d}^k$ are now 1 if $L_v^k \geq d$ and 0 otherwise. With this new variable, we rewrite

$$L_v^k = \sum_{d=1}^{T} x_{v,d}^k \quad \text{and} \quad (L_v^k)^2 = \sum_{d=1}^{T} (2d-1) \cdot x_{v,d}^k,$$

This results in the new objective function

$$\min \quad \frac{1}{D} \sum_{k \in OD} \frac{d_k}{T} \sum_{v \in V^k} \gamma_w \cdot \sum_{d=1}^{T} \frac{1}{2} (2d-1) \cdot x_{v,d}^k + x_{v,d}^k \cdot Y_v^k.$$

This still contains the nonlinear term $x^k_{v,d} \cdot Y^k_v$, which is rewritten as $R^k_{v,d}$ by adding the constraints

$$R^k_{v,d} \leq u^k_v \cdot x^k_{v,d} \tag{4.15}$$
$$R^k_{v,d} \geq l^k_v \cdot x^k_{v,d} \tag{4.16}$$
$$R^k_{v,d} \leq Y^k_v - l^k_v \cdot (1 - x^k_{v,d}) \tag{4.17}$$
$$R^k_{v,d} \geq Y^k_v - u^k_v \cdot (1 - x^k_{v,d}). \tag{4.18}$$

Here, constraints (4.15) and (4.16) force $R^k_{v,d} = 0$ when $x^k_{v,d} = 0$, while onstraints (4.17) and (4.18) ensure that $R^k_{v,d} = Y^k_v$ when $x^k_{v,d} = 1$. To achieve this l^k_v and u^k_v are defined as the lowest and highest possible value that Y^k_v can attain. This makes sure that the first two constraints still work when $x^k_{v,d} = 1$ and the last two constraints when $x^k_{v,d} = 0$.

4.2.2 Minimization over time slice lengths

The linearization of the minimization over the length L^k_v of time slice S^k_v can be achieved by replacing the original constraint with:

$$L^k_v \leq \Delta_{v',v} \tag{4.19}$$
$$\sum_{v \in V^k} L^k_v = T. \tag{4.20}$$

Here constraint (4.19) ensures that the time slices do not exceed the minimum. Constraint (4.20) ensures that the total sum of the lengths L^k_v of the time slices is equal to the time period T. This forces the lengths to be equal to the minimums, since their total sum to T.

4.2.3 Minimization over routes

Finally the minimization of the perceived travel time Y^k_v for passengers of time slice S^k_v is linearized by rewriting the constraint as three linear constraints:

$$Y^k_v \leq Y_r + \gamma_w \cdot \Delta_{v,v'} \tag{4.21}$$
$$Y^k_v \geq Y_r + \gamma_w \cdot \Delta_{v,v'} - M^k_v \cdot (1 - z^k_{v,v',r}) \tag{4.22}$$
$$\sum_{v' \in V^k} \sum_{r \in R^k_{v'}} z^k_{v,v',r} = 1. \tag{4.23}$$

Here, the newly introduced variable $z^k_{v,v',r}$ is 1 if route r is used with departure event v' for time slice S^k_v and 0 otherwise. Constraint (4.21) ensures that Y^k_v is at most the minimum travel time over all routes r with departure v'. Constraints (4.22) requires Y^k_v to be at least as long as the travel time of the chosen route. constraint (4.23) ensures that exactly one route is selected. M^k_v is a constant chosen to be the difference between the longest

and shortest possible route plus the maximum waiting time. This ensures that Y_v^k can be lower than the perceived travel time of a route, which is not the shortest route in constraint (4.22).

$$M_v^k = \gamma_w \cdot T + \max_{r \in R^k} \overline{Y_r} - \min_{r \in R^k} \underline{Y_r}.$$

This ensures that Y_v^k can be lower than the perceived travel time of a route, which is not the shortest route. We note that the maximum waiting time needs to be T instead of $T-1$. This is because the departures are ordered, and thus the time difference between two departures that depart at the same can be T.

4.2.4 MILP formulation

All these linearizations finally result in the following MILP formulation for SPOT.

$$\begin{aligned}
\min \quad & \frac{1}{D} \sum_{k \in OD} \frac{d_k}{T} \sum_{v \in V^k} \sum_{d=1}^{T} \frac{\gamma_w}{2} (2d-1) \cdot x_{v,d}^k + R_{v,d}^k \\
\text{s.t.} \quad & y_{ij} = \pi_j - \pi_i + T p_{i,j} && \forall (i,j) \in A \\
& l_{ij} \leq y_{ij} \leq u_{ij} && \forall (i,j) \in A \\
& Y_r = \sum_{i,j \in r} (y_{ij} + \gamma_t \cdot 1_t(i,j)) && \forall r \in R \\
& \Delta_{v,v'} = \pi_{v'} - \pi_v + T a_{v,v'} && \forall v, v' \in v^k \\
& L_v^k \leq \Delta_{v',v} && \forall k \in OD, v, v' \in V^k, v \neq v' \\
& \sum_{v \in V^k} L_v^k = T && \forall k \in OD, v \in V^k \\
& a_{v,v'} + a_{v',v} = 1 && \forall k \in OD, v, v' \in V^k, v \neq v' \\
& L_v^k \geq d \cdot x_{v,d}^k && \forall k \in OD, v \in V^k, d \in \{1, \ldots, 60\} \\
& L_v^k \leq d \cdot (1 - x_{v,d}^k) - 1 && \forall k \in OD, v \in V^k, d \in \{1, \ldots, 60\} \\
& Y_v^k \leq Y_r + \gamma_w \cdot \Delta_{v,v'} && \forall k \in OD, v, v' \in V^k, r \in R_v^k \\
& Y_v^k \geq Y_r + \gamma_w \cdot \Delta_{v,v'} - M_v^k \cdot (1 - z_{v,v',r}^k) && \forall k \in OD, v, v' \in V^k, r \in R_v^k \\
& \sum_{v' \in V^k} \sum_{r \in R_{v'}^k} z_{v,v',r}^k = 1 && \forall k \in OD, v, v' \in V^k, r \in R_v^k \\
& R_{v,d}^k \leq u_v^k \cdot x_{v,d}^k && \forall k \in OD, v \in V^k, d \in \{1, \ldots, 60\} \\
& R_{v,d}^k \geq l_v^k \cdot x_{v,d}^k && \forall k \in OD, v \in V^k, d \in \{1, \ldots, 60\} \\
& R_{v,d}^k \leq Y_v^k - l_v^k \cdot (1 - x_{v,d}^k) && \forall k \in OD, v \in V^k, d \in \{1, \ldots, 60\} \\
& R_{v,d}^k \geq Y_v^k - u_v^k \cdot (1 - x_{v,d}^k) && \forall k \in OD, v \in V^k, d \in \{1, \ldots, 60\} \\
& L_v^k \in [0, T] && \forall k \in OD, v \in V^k \\
& Y_r \in \mathbb{R}^+ && \forall r \in R
\end{aligned}$$

$$Y_v^k \in \mathbb{R}^+ \qquad \forall k \in OD, v \in V^k$$
$$\pi_v \in \{0, 1, \ldots, T-1\} \qquad \forall v \in V$$
$$p_{ij} \in Z^+ \qquad \forall (i,j) \in A$$
$$a_{v,v'} \in \{0,1\} \qquad \forall k \in OD, v, v' \in V^k, v \neq v'$$

Since SPOT is a minimization problem, it is possible to remove the constraints that enforce upper bounds on $R_{v,d}^k$ and Y_v^k for slightly reducing the size of the model.

4.3 Challenges with the MILP formulation

This MILP formulation for SPOT contains a large number of variables and constraints. As a result, only very small problem instances, such as those involving around 3 train services, can be solved within a reasonable time frame. However, even a small increase in the size of the data input can lead to days of computation time. This means that in most cases, an optimal solution cannot be found within a reasonable amount of time. For larger problem instances involving dozens of train services, it is possible that the solver will stop computing, since it takes excessive amounts of memory for the solvers to run their algorithms.

Another significant problem with the MILP formulation is that even finding an initial solution is challenging for the solver. While, this initial solution is often just setting the first departure of a train service at 0 with the drive and dwell times as short as possible. This solution can always be easily produced instantly along with its objective value. Because of this, Polinder et al. [7] used partly relaxed versions of the MILP to calculate initial solutions. These relaxations are based on the fact that only the PESP constraints need to be satisfied for a timetable to be feasible. The results for this can be found in Chapter 3.5.2 of Polinder's thesis [7]. However, even these results are based on using only a small selection of important OD-pairs.

Finally, we consider the difference between the best found solution and the best found lower bound. In general, a timetable does not need to be optimal to perform well. Other timetables that are close to the optimum can also be effective. However, in Chapter 3.5.2 of Polinder's thesis [7], a very large difference can be observed between the current best found solution and the best found lower bound for SPOT. This significant gap makes it unclear whether the solutions are good or not.

These issues with the MILP formulation motivate our research for fast, reliable and scalable heuristic algorithms.

Chapter 5

Heuristics for SPOT

This chapter will focus on heuristic approaches to find effective timetables for SPOT. Initially, Local Search is explained for finding local optimal timetables. Next, Simulated Annealing is considered to avoid getting stuck in local minima. Following that, lower bounds for SPOT will be established, as this is necessary to assess how close a timetable is to the optimal timetable.

5.1 Local Search

Local search algorithms are heuristics designed to improve an initial timetable step by step to achieve a better result. From this point forward, Local Search is defined as a variant of local search algorithms. Local Search involves examining a neighborhood closely related to the current timetable and selecting the best timetable within this neighbourhood. A new neighborhood is then evaluated for the selected timetable, and the process repeats. The search terminates if no further improvements can be made within any neighborhood of a timetable. At this point, the timetable is the outcome of Local Search. However, since the algorithm only guarantees optimality within the defined neighborhoods, there is no guarantee that a global optimum is found.

5.1.1 Construction of initial timetable

For Local Search there is an initial timetable before neighborhoods can be used to produce better timetables. It is preferred that this initial timetable already contains some sort of structure. For a train service, in many cases it is better to perform the service as fast as possible. This is because a large part of the passengers have direct travel options and therefore the drive and dwell activities are both set at their lower bounds. It is also preferable for a train line with a frequency of 2 or higher that the train services of the line in the same direction are not to close to each other. That is why, an even spread is used for the first departure event times of these train services. This spreading automatically applies to all other events of the train services, since the first preference structure equalizes the drive and dwell times of the services.

An initial timetable is constructed by first randomly assigning initial departure times to each train service. The subsequent departure and arrival times are then calculated for each train service based on the shortest drive and dwell times after the initial departure. To achieve an even spread for train lines with a frequency of 2 or higher, the arrival and departure times of one train service are used as the basis, the so-called "basis times". the other train services are then scheduled by shifting these basis times to ensure an even distribution. For a frequency of 2, a shift of 30 minutes is applied, while for a frequency of 3, one train service is shifted by 20 minutes and another by 40 minutes.

5.1.2 Types of neighborhoods

For Local Search, the neighborhoods need to be defined. In the case of SPOT, there are no infrastructural constraints between different train services. Therefore, a train service can be modified without impacting other train services, making it a good basis for the neighborhoods. Three types of neighborhoods for a train service are defined, with an example of a timetable out of these neighborhoods for the train service in Figure 5.1. These three types of neighborhoods are defined by train service shifts, dwell time shifts and drive time shifts.

Figure 5.1: event-activity network of a single train service

- **Train service shift:** Train service shifts define a neighborhood containing all timetables where a shift is applied to a complete train service. A shift of a train service involves increasing or decreasing all event times of the train service by the same number of minutes. The neighborhood consists of all 60 possible shifts on the train service, including a shift of 0 minutes. For example, a shift of 3 minutes for the trains service in Figure 5.1 results in the train service in Figure 5.2. Each train service has one such neighborhood.

Figure 5.2: Event-activity network of a train service with a train service shift of 3 minutes.

Train service shifts have clear influence on the initial waiting time and the transfer time of passengers. First, By shifting a train service, The spreading of the departures at

all stations of the train service are changed. With this a good spreading of departures in the same direction can be made. For example, if there are 6 trains services in the same direction then a train service every 10 minutes is good. However, there can be frequency changes on the route of the train service where 2 trains go in another direction. This leaves 4 trains, for which a train service every 15 minutes is good. It is often not possible to keep both the 10 minute spread at one station and the 15 minute spread at the next station of a train service. Lastly, Shifting a train service influences transfers, by increasing or decreasing the transfer time to or from the train service at a station. However, This change impacts the whole train service and thus an improvement of a transfer can result in a worse transfer at another station.

- **Dwell time shift:** Dwell time shifts define a neighborhood containing all timetables where the dwell time of a train service at a station is changed. This change can be achieved by shifting all events before or after the dwell activity of a train service. For example, the dwell time of the train service at station S_2 in Figure 5.1 can be extended by 3 minutes by shifting all events after the dwell activity by 3 minutes, resulting in the train service in Figure 5.3. This neighborhood consist of all possible timetables created by shifting all events before or after the dwell activity, as long as the dwell constraint is still valid. These neighborhoods exist for every dwell activity of every train service.

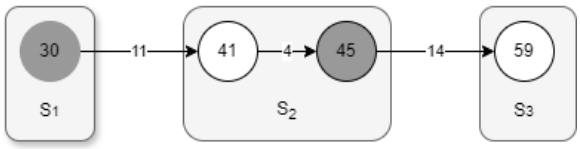


Figure 5.3: Event-activity network of a train service with a dwell time shift.

Dwell time shifts impact various aspects of a passenger travel time, including initial waiting time, The changes in initial waiting time and transfer time due to dwell time shifts are similar to those caused by shifting a train service but are applied to a smaller segment of the train service. This smaller effect makes it easier to optimize departure spreading and transfer times, as fewer stations are affected. Additionally, dwell time shifts also influence the in-train time for passengers who remain on the train during the altered dwell activity. Since a significant portion of passengers use direct connections between their start and end stations, shorter dwell times are generally preferred. This is because shorter dwell times reduce the in-train time for these direct passengers, improving overall travel efficiency.

- **Drive time shift:** Drive time shifts define a neighborhood containing all timetables where the drive time of a train service between stations is changed. These drive time shifts are defined in the same way as the dwell time shift, but now by shifting all

events before or after the drive activity. For example, the drive time of the train service from station S_1 to station S_2 in Figure 5.1 can be extended by 3 minutes by shifting all events after the drive activity by 3 minutes, resulting in the train service shown in Figure 5.3. These neighborhoods are defined for all drive activities of all trains services and are bounded by the drive constraints.

Figure 5.4: Event-activity network of a train service with a drive time shift.

Drive time shifts operate similarly to dwell time shifts applied to the subsequent dwell time. For the initial waiting time and the transfer, the effect are identical. However for the in-train time, the shift affects not only passengers continuing through the station, but also those arriving at the station. Consequently, an increase in drive time is always worse than an increase in the next dwell time, and a decrease in drive time is always better than a decrease in the next dwell time. This means that a drive time shift is only beneficial if the next dwell time is already maximized or the drive time is not minimized. Therefore, drive time shift are rarely beneficial, since the initial timetable starts with all drive times as short as possible.

It can be useful to use a neighborhood where multiple train services are adjusted together. For instance, in the case of a train line with a frequency of 2, applying the same shift to both train services in the same direction can be beneficial. This approach helps to maintain or find a good spread of departures, keeping the initial waiting time low. Another possibility is ensuring the spreading of all trains at a station in a particular direction to keep the initial waiting time at low. However, this approach affects many train services and therefore represents a rather extreme change.

In Local Search, all possible neighborhoods are compiled into a list, which dictates the order in which neighborhoods are explored to find better results. After checking all neighborhoods in the list, the process returns to the first neighborhood and continues iterating. Local Search terminates when every neighborhood has been examined and no further improvements can be made. Using a predetermined order for neighborhoods ensures that each neighborhood is considered approximately the same number of times, promoting a thorough exploration of potential timetables.

5.2 Simulated Annealing

Simulated Annealing is a variant of Local Search where a single timetable is chosen from a neighborhood of a timetable and accepted with a certain probability. This probability is always positive, allowing even worse timetables than the current timetable to be accepted. As a result, the algorithm can occasionally choose a worse timetable over a local minimum. If such a worse timetable is accepted, a new neighborhood is generated around it. This new neighborhood differs from that of the local minimum, providing a chance to discover a better timetable than the local minimum. Thus, Simulated Annealing can escape local minima, improving the chances of finding better local minima or even the global minimum.

5.2.1 Probability of accepting chances

For Simulated Annealing, implementing appropriate probabilities for accepting a new timetable is crucial. If these probabilities are not well-calibrated, there is a risk that worse timetables may either be accepted too easily, leading to suboptimal timetables, or almost never be accepted, hindering the algorithm's ability to escape local minima. Therefore, Simulated Annealing uses an acceptance probability that is 1 if the new timetable is better, and the following chance if the new timetable is worse:

$$\mathrm{e}^{-\frac{E(\pi)-E(\pi')}{T}}.$$

Here $E(\pi)$ and $E(\pi')$ represent the perceived travel times of the current timetable π and the new timetable π', respectively. The temperature T affects the acceptance probability, with higher temperatures increasing the likelihood of accepting worse solutions. This function ensures that the change of accepting a worse timetable decreases exponentially with the difference in perceived travel times between the current and new timetable.

At the start of Simulated Annealing, the timetable is still quite random, so there is a high chance that a change will improve the timetable. However, as the algorithm progresses and better timetables are found, the likelihood of selecting worse timetables in the neighborhood increases. This can lead to a situation where the chance of selecting and accepting a worse timetable becomes higher than the chance of selecting a better timetable. To address this, the temperature is decreased, making it less likely that worse timetables will be accepted. This means that instead of a fixed temperature, an initial temperature $T_{initial}$ is required. This initial temperature should be carefully chosen to balance the search process: it should not be too low, as this would prevent the algorithm from escaping local minima, nor should it be too high, as this would require extra time for the temperature to decrease before the timetable begins to improve.

The decrease in chance of accepting worse timetables for Simulated Annealing is achieved by lowering the temperature T. A lower temperature exponentially decreases the likelihood of accepting a worse timetable. Here, the temperature decrease δ_T is a factor smaller than 1, which updates the temperature by $T_{new} = T_{old} \cdot \delta_T$ every time a worse timetable is accepted.

This factor needs to be close to 1 to facilitate escaping local minima, while still being low enough to ensure that good timetables are found in a reasonable time.

5.2.2 Choosing a timetable from a neighborhood

For Simulated Annealing it is crucial to define how a timetable is chosen from a neighborhood of the current timetable. This is either done by choosing a random timetable from all possible timetables in the neighborhood or by assigning to each timetable in the neighborhood a probability of being selected and then choosing one based on these probabilities. In the case of timetables, it is slightly known which timetables are more likely to improve the perceived travel time. Therefore, the probabilities of choosing a timetable can be higher for those that are more likely to improve travel time, giving a preferred direction to Simulated Annealing while still allowing all options in the neighborhood to be considered.

In Simulated Annealing, the probabilities for choosing timetables within neighborhoods created by train service shifts and dwell time shifts are distributed differently. For a train service shift, smaller shifts have more in common with the initial timetable than larger shifts. Since the current timetable is already partly optimized, checking closely related timetables is often more beneficial. Therefore, the chances of choosing a timetable based on a train service shift will follow a normal distribution centered around a shift size of 0. In the case of dwell time shifts, it can also be argued that smaller shifts are preferable. However, as previously noted, shorter dwell times are generally more beneficial because they improve the experience for direct passengers. Therefore, shifts that shorten dwell times should be given a higher probability. After selecting a shift, there is an 50/50 chance of choosing to shift all events before or after the dwell activity.

5.3 Lower bounds on the perceived travel time of a timetable

Lower bounds for the heuristically found timetables are used to evaluate the quality of a timetable and identify potential issues. Calculating these lower bounds is challenging, since it is hard to determine if transfer times can be as short as possible due to conflicting passenger preferences. These different preferences also apply on the initial waiting time of passengers at their start station. For simplicity, the focus of the lower bound is on achieving a well-distributed spread of departure times and short perceived travel times with minimal transfer times.

For all following lower bounds, a preselected set of routes per OD-pair is used to reduce the problem size, which means there is no guarantee of achieving a lower bound on the perceived travel time. However, since the preselected routes are expected to contain the best route options for an OD-pair, it is expected that the lower bounds do not significantly differ from those obtained by considering all possible routes.

5.3.1 Simple lower bounds per OD pair

The lower bounds are based on individual lower bounds per OD-pair rather than for all OD-pairs together. This is done because assuming that all transfer times are as short as possible significantly reduces the influence of conflicting passenger preferences, leaving only initial waiting, which influences only passengers from the same start station. Now, the sum of the lower bounds for each OD-pair, multiplied by the number of passengers for that OD-pair, provides a lower bound for all OD-pairs collectively.

A very simple lower bound per OD-pair is defined by ignoring the perceived travel time for waiting at the start station. This requires determining the shortest perceived travel time for the OD-pair by taking the minimum over the perceived travel times of all possible routes for the OD-pair, assuming that all drive, dwell, and transfer activities are as short as possible. This shortest perceived travel time is a lower bound on the average perceived travel time for the OD-pair, as no passenger can have a shorter perceived travel time.

By including initial waiting times, a better lower bound is found for every OD-pair. This involves first identifying how many different train services can serve as the initial departure from the start station given the possible routes for the OD-pair. Assuming that all these departures have a perceived travel time equal to the previously determined shortest perceived travel time, an optimal spread of departures is created. This is achieved by evenly distributing the departures over the hour, resulting in a departure every $\frac{60}{\#D}$ minutes, where $\#D$ is the amount of different initial departures. This approach yields a lower bound of the shortest perceived travel plus an average waiting time of $\frac{30}{\#D}$

A better lower bound is established by considering the shortest route for an OD-pair per departing train service. This is useful since the shortest perceived travel time from a specific initial departure is longer or equal to the shortest perceived travel time across all initial departures, resulting in a better lower bound. When different departures have varying shortest perceived travel times, evenly distributing departures may no longer be optimal. Instead, it may be advantageous to have a better spread of departures with a shorter perceived travel time. For this we search for the optimal distribution of departures across the hour for the passenger intervals.

Theorem 5.1. *Consider x_j as the length of the time slice of passengers that take route j with perceived travel time of ℓ_j. Let L represent the length of a single time interval of passengers and γ_w denote the waiting penalty. We search for $x_1,\ldots,x_n$ with $x_j \geq 0$ for all j and $\sum_j x_j = T$ such that*

$$f(x) = \sum_j (0.5 \cdot x_j \cdot \gamma_w + \ell_j) \cdot x_j = \sum_j L \sum_{i=1}^{x_j/L} ((i - 0.5) \cdot L \cdot \gamma_w + \ell_j)$$

is minimized (0.5 follows from the preferred departure times in the middle of the intervals

of passengers). This is solved to optimality by the following algorithm where $T_{ininitial}$ is the initial amount of intervals:

- *Start by setting $x_1 = \ldots = x_n = 0$ and $T = T_{initial}$.*
- *Choose route j with the lowest $x_j + \ell_j$ and increase x_j by L, representing extra real waiting time for the next passenger interval and decrease T by L.*
- *Repeat the last step until T is 0.*
- *Now $\sum_j x_j = T_{initial}$, $x_j \geq 0$ for all j and $(x_j \cdot \gamma_w + \ell_j) - (x_i \cdot \gamma_w + \ell_i) \leq \gamma_w \cdot L$ for all j, i if $x_j > 0$ and $x_j \cdot \gamma_w + \ell_j \leq \ell_i$.*

Proof. All solutions with this structure have the same objective value as the only difference is in which routes get the the longest perceived travel in the last steps. This longest perceived travel time for the last passengers is the same independent of the route and therefore results in the same objective value.

Given an arbitrary $x_1,\ldots,x_n$ with with $x_j \geq 0$ for all j, $\sum_j x_j = T_{initial}$ and there exist routes i and j such that

$$(x_j \cdot \gamma_w + \ell_j) - (x_i \cdot \gamma_w + \ell_i) > \gamma_w \cdot L \text{ for } x_j > 0.$$

Then $(x_j \cdot \gamma_w + \ell_j) > ((x_i + L) \cdot \gamma_w + \ell_i)$, meaning that increasing x_i by one interval and decreasing x_j by one interval lowers the perceived travel time of the interval of passengers while keeping all other perceived travel times the same. Repeating this leads to an spreading with

$$(x_j \cdot \gamma_w + \ell_j) - (x_i \cdot \gamma_w + \ell_i) \leq \gamma_w \cdot L \text{ for all } j, i \text{ if } x_j > 0 \text{ and } x_j \cdot \gamma_w + \ell_j \leq \ell_i.$$

All solutions with this structure have the same objective value and are therefore optimal. □

5.3.2 Restrictions on the routes

The following lower bound relies on that there is no extra waiting time penalty. Therefore the perceived travel time per minute waited at the start station is set equal to 1. This is the penalty used in this thesis and the exact reasoning for this penalty value is further explained in Chapter 6.

For an OD-pair the shortest route of two different initial departures can use the same segment of a train service after a transfer, however, often one of these routes is not useful. Consider three stations S_1, S_2 and S_3 with train services L_1, L_2 and L_3 as illustrated in Figure 5.5. For the OD-pair from start station S_1 to end station S_3, the shortest route with

departure on L_1 involves taking L_1 and then L_3, while the shortest route for a departure on L_2 involves taking L_2 and then transferring to L_3. In this scenario, taking the last departure from S_1 that still allows for the earliest transfer to L_3 at S_2 is always better or at least equal to taking the other departure, since waiting at the start station costs the same amount of perceived travel time as waiting in the train or at a transfer. Therefore, only one of these shortest routes is necessary.

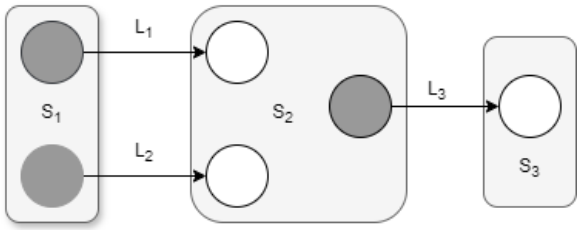


Figure 5.5: Train network of three train services L_1, L_2 and L_3 over stations S_1, S_2 and S_3.

It is important to note that there are exceptions where both routes are important. This is the case when one route has more transfers than the other before the same train service is used. For instance, consider Figure 5.6 with two possible routes: taking train service L_1 then L_4 and then L_3 or taking L_2 and then L_3. In this case taking L_1 is the last departure, which still makes the earliest transfer at S_2, however this route has extra perceived travel minutes due to an extra transfer at S_4. This results in that taking L_2 is faster for all passengers with preferred departure time in (1,60], while for passengers in (0,1] the extra transfer penalty should be less than waiting 59 more minutes. This means that both routes are used by different passengers, even though they use the same train service.

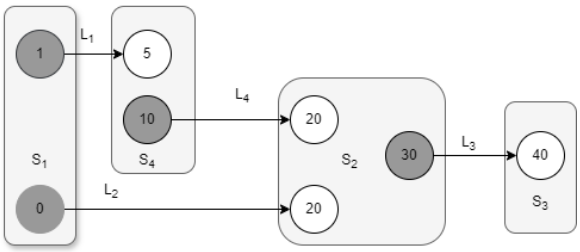


Figure 5.6: Train network of three train services L_1, L_2, L_3 and L_4 over station S_1, S_2, S_3 and S_4 with departure times.

An improved lower bound is established by evaluating all combinations of assigning a route to every initial departure for an OD-pair. If two routes use the same train service with the same number of transfers before taking the same train service, then the combination is ignored. By replacing one of these routes, the shared train service can be avoided, resulting

in a longer perceived travel time over the route, as the new route cannot be shorter than the original shortest route. For every route-departure combination of an OD-pair, an optimal departure spreading is calculated by Theorem 5.1 to determine the corresponding perceived travel time. The lower bound for the OD-pair is then defined as the minimum perceived travel time across all these combinations.

Chapter 6

Data acquisition

This chapter explains the sources of all data, the methods used to process it for producing input data for SPOT, and the rationale behind certain data choices. The following four input data types are considered: the train network, the OD-matrix, the SPOT penalty parameters, and the preselected routes.

6.1 Train network

The initial train network data is obtained from an initial hourly weekend timetable for the start of 2024. This hourly-based PESP problem operates with times measured in one-tenth of a minute. The initial train network data comprises two files: one detailing all train services with assigned times for each event, and the other listing all constraints between events along with their types.

The constraints in this initial data consist of seven different types: drive, dwell, dwell without stop, headway, frontal collision, crossing, and turnaround constraints. For SPOT, only the drive and dwell constraints are necessary, and all others are removed. The dwell without stop constraints at a station are combined with the drive constraints to and from that station of the same train service, forming a new drive constraint for the direct connection between the preceding and succeeding stations. Additionally, transfer constraints are added, with a uniform lower bound of 3 minutes for all transfers. This assumption is made due to the lack of specific infrastructure data, allowing all transfers to be treated as cross-platform transfers, which can always be accomplished within 3 minutes.

The initial data contained some inconvenient structures for strategic timetabling that needed addressing. Specifically, there were cases where two train services of the same line and direction had differing drive or dwell constraints at the same stations. Given that these services follow the exact same route, no differences in constraints should exist in the early stages of planning. To resolve this, the minimum of the lower bounds was set as the new lower bound, and the maximum of the upper bounds was set as the new upper bound. Another inconvenient part of the data involved train services within the same line having different paths, this occurred when a train line with a frequency of 2 per hour had one train

service that only covered part of the complete path. In such cases, a new train line was created to split the train services to make sure that all train services of a train line follow the same path.

For this thesis is worked with a subset of the 384 train services from the initial data to reduce the calculation times of the heuristics. This subset includes all intercity trains, comprising of 25 train lines with a total of 80 train services. This subset was chosen because these train services are designed to provide optimal connections between all major stations.

6.2 OD-matrix

NS has its own OD-matrices detailing the number of passengers traveling between stations per hour per day; however, these matrices are confidential. Consequently, we need to construct an synthetic OD-matrix using publicly available data. Available data includes the number of check-ins and check-outs per station, as well as the rail distances in kilometers between all stations. By leveraging these two data sources, a synthetic OD-matrix is created that approximates the actual OD-matrix.

With the available data, two steps are performed to create the synthetic OD-matrix. The first step involves making an initial estimate for each OD pair and the second step adjusts the matrix to ensure that it is symmetric and the total number of passengers is consistent with the check-ins and check-outs. For convenience, the number of check-ins and check-outs at a station will be referred to as the station demand, which is the number of passengers using the station.

- First, an initial matrix is created based on a simple guess for the number of passengers for every OD-pair. This guess is determined by considering the station demand of both stations and the distance over rail between the two stations. It is expected that more passengers travel between stations with high station demands and between stations that are close to each other. The initial number of passengers for an OD-pair is defined as:

$$d_{A,B} = \frac{(P_A \cdot P_B)^{1.3}}{max(40, dist(A,B))^{1.3}}.$$

  Here A and B are the start and end station with $d_{A,B}$ as the passenger demand for the OD-pair, P_A and P_B as the station demands and $dist(A,B)$ as the distance between the station over rail. Both the numerator and the denominator are raised to the power of 1.3, as it is expected that station demand and distance have more than a linear influence. The maximum in the denominator is used to ensure that the passenger demand between two stations that are close to each other is not overestimated by using a minimum of 40 km. In the case of stations that are close to each other within the same city, the passenger demand is set to 0. This is done because, in many cases, traveling between two stations in the same city is not typically done by intercity trains.

The cities considered for this adjustment are Amersfoort, Amsterdam, Den Helder, Den Haag, and Rotterdam, each containing multiple intercity stations.

- The second step ensures that the matrix is symmetric and that the station demand equals the sum of passenger demands from the station to any other station. The matrix is made symmetric because most passengers eventually return to their original starting points, representing an average OD-matrix over a day. However, ite needs to be noted that the OD-matrix is not always symmetric during specific times, such as morning and evening rush hours, when passenger demands to and from work are higher. To achieve a symmetric OD-matrix with the correct passenger counts per station, a convex quadratic formulation is used. This approach minimizes the difference between the initial passenger demands from the first step and the newly adjusted passenger demands, ensuring accuracy and balance in the timetable

  The initial passenger demands of the first step only represents the relative number of passengers between stations. Because of this, the correlation between the derived OD-matrix and the initial matrix is not a good comparison. To address this, the passenger demands for departures from each station will be scaled so that the sum matches the station demand for that station. This adjustment immediately satisfies the constraint that the station demand equals the total passenger demand departing from the station. However, this adjustment disrupts the symmetry of the passenger demands. Consequently, an convex quadratic formulation is still required to restore the symmetry while maintaining the adjusted demands.

  The convex quadratic problem is formulated as

$$\min \quad \left(\frac{x_{i,j} - d_{i,j}}{(d_{i,j})^{\alpha}}\right)^2 \tag{6.1}$$

$$x_{i,j} = 0 \qquad \forall i, j \in S, d_{i,j} = 0 \tag{6.2}$$

$$\sum_{j \in S} x_{i,j} = S_i \qquad \forall i \in S \tag{6.3}$$

$$x_{i,j} = x_{j,i} \qquad \forall i, j \in S \tag{6.4}$$

$$x_{i,j} \in \mathbb{R}^+ \qquad \forall (i,j) \in A \tag{6.5}$$

  Here is $d_{i,j}$ the initial passenger demand and S_i the publicly available data of passengers per station. Constraint (6.2) ensures that the new passenger demands x is zero if the initial demand d is zero. Constraint (6.3) enforces that the sum of the passenger demands from a station i equals the station demand S_i. Constraint (6.4) ensures symmetry and constraint (6.5) ensures that all passenger demands are non-negative. This problem can be infeasible depending on the values S_i, however this is not the case here. The objective function minimizes the quadratic difference between the new and initial passenger demands. The factor $(d_{i,j})^{\alpha}$ is introduced to fix the type of difference, with $\alpha = 0$ representing a linear difference and $\alpha = 1$ representing

a percentage difference. Setting $\alpha = 0$ emphasizes large demands, potentially forcing small demands to zero, while $\alpha = 1$ emphasizes small demands, possibly creating significant differences for the larger demands which represent the most important station connections. Ultimately, $\alpha = 0.4$ is chosen as a compromise, ensuring the value is low enough to keep almost all demands greater than zero while balancing the importance of both small and large demands.

The final OD-matrix is reduced to make calculations faster. For this, 10 percent of all passengers are removed from the OD-matrix. This is done by iteratively setting the smallest OD-pair to 0 passengers until 10 percent of the passengers have been removed. It is chosen that if the passengers demand from station A to station B is removed, then also the passenger demand from station B to station A is removed, since these two passenger demands are originally the same by symmetry.

6.3 Penalty parameters

The waiting penalty penalizes every waiting minute at the starting station. This penalty was introduced in SPOT to spread departures evenly for passengers. However, there is chosen to set the penalty at 1, meaning no extra perceived travel time. One reason for this choice is that waiting at a transfer is equivalent to waiting at the start station, which is not penalized per minute. Another reason is that a penalty higher than 1 can lead to illogical routes being taken by passengers. For example consider the event-activity network in Figure 6.1. If a passenger wants to go form station S_2 to S_1 and has a preferred departure time at 56, then the logical option is the direct train at 39. However, with an extra waiting penalty, the passenger can go from S_2 to S_3 and then from S_3 to S_1. This route is taken if the extra waiting penalty on the 43 minutes of waiting is higher than the transfer penalty at S_3. This route includes visiting station S_2 twice, which is almost certainly not a route any passenger would logically take and Thus contradicts the passenger oriented approach of evaluating the timetable in SPOT.

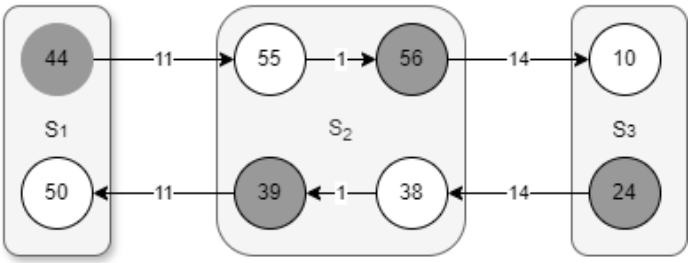


Figure 6.1: Event-activity networks of 1 train line.

The transfer penalty is introduced to penalize every transfer in a route between stations. De Keizer et al. [5] extensively researched how passengers perceive these transfers. They

found that, on average, a transfer is perceived as taking 23 minutes, plus an additional value based on the extra transfer time. Since a minimum transfer time of 3 minutes is used, the transfer penalty is set at 20 minutes per transfer.

6.4 Preselected routes

The preselected routes are chosen to ensure a high likelihood that the routes, which passengers take for a good timetable are included. Initially, a set of routes is created by considering all possible routes with a maximum of two transfers. Limiting to two transfers ensures that there is at least one feasible route between every start and end station. These routes are calculated by first identifying all possible direct routes, then expanding these direct routes by adding another direct route to form routes with one transfer. Finally, these one-transfer routes are expanded further by adding another direct route to form routes with two transfers.

These initial routes are then evaluated in four steps to filter out the routes that are least likely to be used by passengers:

1. The first step consists of removing routes that visit a station twice. If such a route is included, there is also a route that directly transfers at the station with a lower or equal perceived travel time. This is because waiting at a station and waiting in a train cost the same amount of perceived time per time unit.

2. The second step involves removing unnecessary transfers. For each route, every transfer will be checked, and if at least one transfer is deemed unnecessary, the route is removed. A transfer is considered unnecessary when the following condition is met: the train service before the transfer can directly take the passenger to the end station or the train service after the transfer already starts at the start station and the added transfer is not fast enough to overcome an extra 23 perceived minutes. The 23 minutes is based on the extra perceived travel time from the transfer penalty (20 minutes) and the minimum transfer time (3 minutes).

3. The third step involves comparing routes for the same OD-pairs by their expected and minimum perceived travel times. Here, the expected travel time is defined by adding the expected drive, dwell, transfer and initial waiting time of a route. For expected drive and dwell times, the minimum values are chosen. For a transfer, the expected time is the 23 minutes of the transfer penalty and minimum transfer time plus an extra waiting time of $\frac{30}{|direct_{i,j}|}$ minutes. Here $|direct_{i,j}|$ is the amount of direct train services that go from the transfer station to the next transfer or end station on the route. This is then the expected waiting time for a random arrival time with evenly spread departures of the direct trains. In the case of initial waiting time, $\frac{60}{|direct_{i,j}|}$ minutes is used, which accounts for the passengers arriving just after a departure. These passengers exist since the preferred departure time of passengers is evenly spread over

all possible times. With the expected perceived travel time of a route, the lowest expected perceived travel time for each OD-pair is identified. If a route for the same OD-pair has a minimum perceived travel time higher than this expected perceived travel time, that route is removed from consideration.

4. The fourth step is a refinement of the third step, focusing on comparing routes for the same OD-pairs that also share the same first train service. By isolating routes that begin with the same train service, it becomes clear that passengers will experience the same initial waiting time at the start station. Consequently, when comparing these specific routes, the initial waiting time component in the expected perceived travel time can be ignored.

Chapter 7

Results of heuristics

In this chapter, the results of the heuristics for SPOT are evaluated. Initially the Local Search heuristic is evaluated by observing how changes in the heuristic affect the outcome and speed. Following this, we assess the Simulated Annealing heuristic, focusing on its performance and contrasting it with the Local Search heuristic. Finally is evaluated how the heuristics and problem simplifications contribute to the overall effectiveness of the timetable.

7.1 Results for Local Search

For Local Search, several factors affect the final timetable. This chapter will address three key aspects: the types of neighborhoods, the size of the neighborhoods, and the order in which these neighborhoods are evaluated.

7.1.1 Types of neighborhoods

As previously mentioned in Section 5.1.2, there are three fundamental types of Local Search neighborhoods, which are defined by train service shifts, dwell time shifts and drive time shifts. It was also stated that drive time shifts are generally ineffective, because dwell time shifts provide the same benefits but fewer drawbacks. With train service shifts, it is possible to apply the shift across all train services of a train line in the same direction. However, this approach is not feasible for dwell time shifts due to the constraints that prevent the same dwell shift from being applied across multiple train services. Therefore, train service shifts for a train service and a train line in a single direction are taken into account, as well as individual dwell time shifts.

In Figure 7.1, the results of two Local Search executions are shown. Both executions allow train service shifts; however, one includes dwell time shifts while the other does not. Clearly, including dwell time shifts extends the Local Search process. Without dwell time shifts, a local minimum is found in 1787 seconds, while with dwell time shifts, no local minimum is found even after an hour. This is because including dwell times introduces additional neighborhoods that need to be evaluated for optimality, and there are also more feasible timetables when dwell times are variable. Initially, the improvement over time is

smaller with dwell time shifts. After the first 10 minutes, the heuristic without dwell time shifts achieves a timetable with average perceived travel time of 63.73 minutes, while for with dwell time shifts, an additional 200 seconds are required to reach the same improvement. This is likely because dwell time shifts only change a smaller part of a train service, resulting in smaller improvements. However, while the Local Search without dwell time shifts stops improving at a perceived travel time of 63.27 minutes, the Local Search with dwell time shifts continues to improve, ultimately reaching a perceived travel time of 62.33 minutes. This indicates that incorporating dwell time shifts can significantly enhance the perceived travel time of passengers, offering an improvement of a full minute.

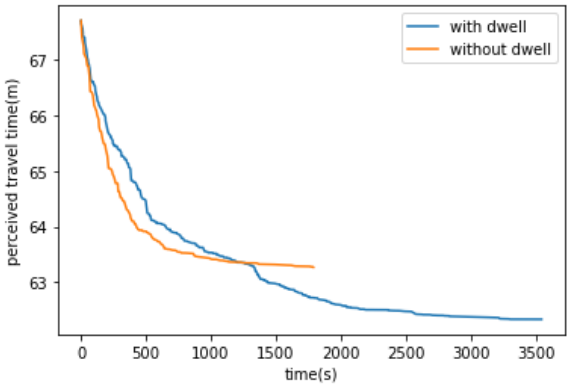


Figure 7.1: Perceived travel time of the best solution found over time for Local Search executions with and without dwell shifts.

7.1.2 Size of neighborhoods

For both train service shifts and dwell time shifts, the possible shifts are defined within specific ranges. Train service shifts can range from 30 minutes earlier to 30 minutes later, while dwell time shifts can vary from the minimum dwell time to the maximum dwell time, with adjustments to event times before or after the dwell time. To streamline the process, it is possible to limit the size of these neighborhoods by setting a maximum shift size. To evaluate the effectiveness of different neighborhood sizes, four Local Search runs were conducted with maximum shifts of 30, 20, 10, and 5 minutes, as shown in Figure 7.2 and Table 7.1. The results show that Local Search with smaller neighborhoods improve faster during the first 20 minutes, although the difference is minimal. This is probably due to the dwell time shifts, where the maximum shifts are already close to 5 minutes. Ultimately, the heuristic outputs are better when using larger neighborhoods, as there is a higher chance of applying the optimal shift. Therefore, it is concluded that for the current cases, a maximum shift of 30 minutes is effective.

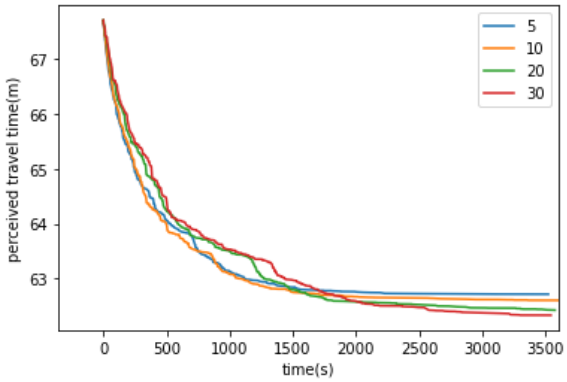


Figure 7.2: Perceived travel time of the best solution found over time for Local Search executions with the different neighborhood sizes of maximum shift of 5, 10, 20 and 30 minutes.

Table 7.1: Perceived travel time of the best solution found after 10, 20, 30, 40, 50 and 60 minutes of the basic Local Search heuristic with a max shift of 5,10,20,and 30 minutes.

	10	20	30	40	50	60
max 5 minutes	63.87	62.97	62.78	62.73	62.72	62.72
max 10 minutes	63.79	62.90	62.70	62.65	62.62	62.60
max 20 minutes	63.97	63.21	62.61	62.54	62.47	62.43
max 30 minutes	64.06	63.36	62.72	62.49	62.38	62.33

7.1.3 Order of neighborhoods

Finally, the impact of the order in which neighborhoods are explored was analyzed. To do this, a list of all possible train shifts and dwell time shifts was created and shuffled to determine the sequence in which neighborhoods are used. The results of five different shuffles are shown in Figure 7.3. Overall, the improvement is quite similar across all five executions, with the five shuffles yielding perceived travel times of 63.38, 63.35, 63.46, 63.44, and 63.34 minutes after 10 minutes. However, the final outputs show a significant variation, with perceived travel times of 62.33, 62.54, 62.63, 62.51, and 62.62 minutes respectively. This difference of up to 0.3 perceived minutes suggests that it is beneficial to perform the Local Search heuristic multiple times with different neighborhood exploration orders to potentially achieve better results.

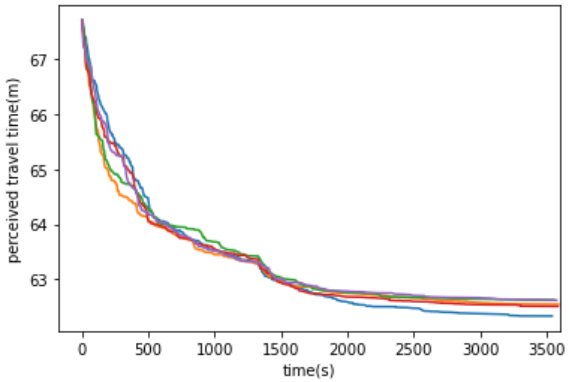


Figure 7.3: Perceived travel time of the best solution found over time for 5 Local Search executions with different orders of neighborhoods.

7.2 Results Simulated Annealing

For Simulated Annealing, three factors are evaluated that influence the final timetable: the types of neighborhoods, the temperature, and the order of neighborhoods. Unlike Local Search, the size of service shifts is not considered in Simulated Annealing, as it examines only one point within the neighborhoods at a time.

First, basic parameters for Simulated Annealing are established. The algorithm is terminated after no changes occur in the last 1000 neighborhoods, as Simulated Annealing continues to search for solutions even after reaching a local minimum. For train service shifts, a normal distribution with a variance of 5 minutes is used to determine the shift. Variances ranging between 1 and 10 minutes were tested, but no significant differences in outcomes were observed. For dwell time shifts, a 20 percent chance is assigned to the smallest possible dwell time, with each additional tenth of a minute having a 20 percent chance of being selected from the remaining probability. Similar to train service shifts, varying these probabilities did not lead to notable differences in the results.

7.2.1 Types of neighborhoods

In a similar manor to Local Search, the impact of including or excluding dwell time shifts in Simulated Annealing is evaluated. For this purpose, Simulated Annealing is applied twice, once with dwell time shifts and another without dwell time shifts, as shown in Figure 7.4. The results indicate that the approach excluding dwell time shifts improves slightly faster. However, after 1 minute. without dwell time shifts has found a timetable with a perceived travel time of 63.32, while with dwell time shifts has found a timetable with perceived travel time of 63.30. After this initial minute, there are still significant improvements with dwell time shifts, while without dwell time shifts the improvement stagnates. Finally, without dwell time shifts outputs a timetable with a perceived travel time of 63.01 minutes, while

with dwell time shifts results in a timetable with a perceived travel time of 62.16 minutes. Consequentially, Simulated Annealing with dwell shifts has a significant improvement compared to without dwell time shifts.

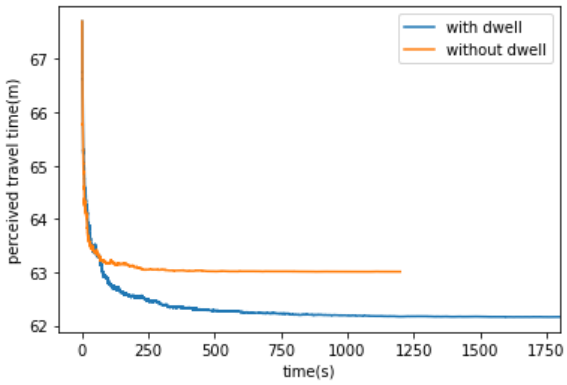


Figure 7.4: Perceived travel time of the best solution found over time for Simulated Annealing executions with and without dwell time shifts.

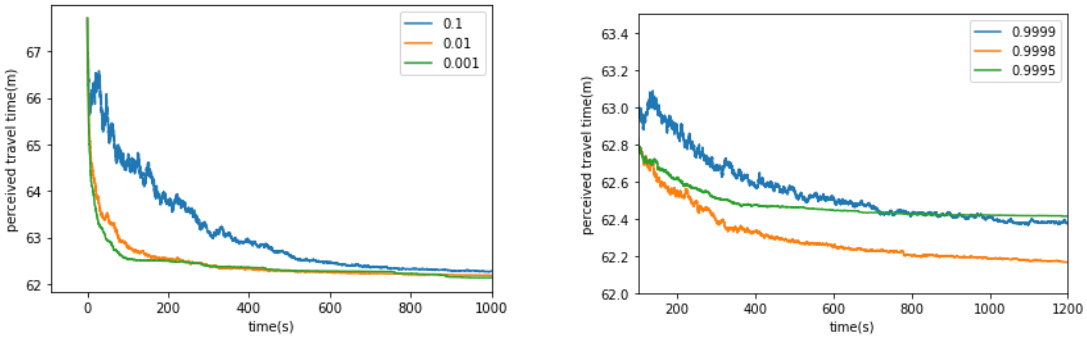
7.2.2 Temperature settings

Simulated Annealing operates on the principle of initially accepting poorer results with some probability by controlling the temperature and its decay. The temperature parameter governs the likelihood of accepting worse solutions during the search process. As the algorithm progresses, the temperature decreases, which is applied whenever a worse result is accepted.

In Figure 7.5a, three Simulated Annealing executions are shown with different starting temperatures: 0.1, 0.01, and 0.001. With a starting temperature of 0.001, only very small increases in travel time are permitted, which makes it challenging to escape local minima. Conversely, a starting temperature of 0.1 allows for large increases in travel time early in the process, but this is unnecessary as most local minima are still significantly better at this stage. The starting temperature of 0.01 strikes a balance between these extremes, allowing small increases in perceived travel time while maintaining enough flexibility to escape local minima.

In Figure 7.5b, three Simulated Annealing executions are presented with different temperature decrease factors: 0.9995, 0.9998, and 0.9999. The first minute is not shown in the figure, because the effectiveness of the temperature decrease at escaping local minima becomes more apparent after this initial period. With a decrease factor of 0.9995, the temperature drops too quickly, leading to only small increases in perceived travel time, making it unlikely that local minima are escaped. In contrast, a factor of 0.9998 allows for occasional increases of perceived travel time, followed by noticeable improvements in perceived

travel time. With a factor of 0.9999, significant changes in timetables continue to occur later in the process. This extended exploration delays the discovery of good timetables, but it also increases the likelihood of finding better solutions compared to faster temperature decreases. Due to computational time considerations, a factor of 0.9998 is ultimately chosen, striking a balance between effective exploration and efficient computation.



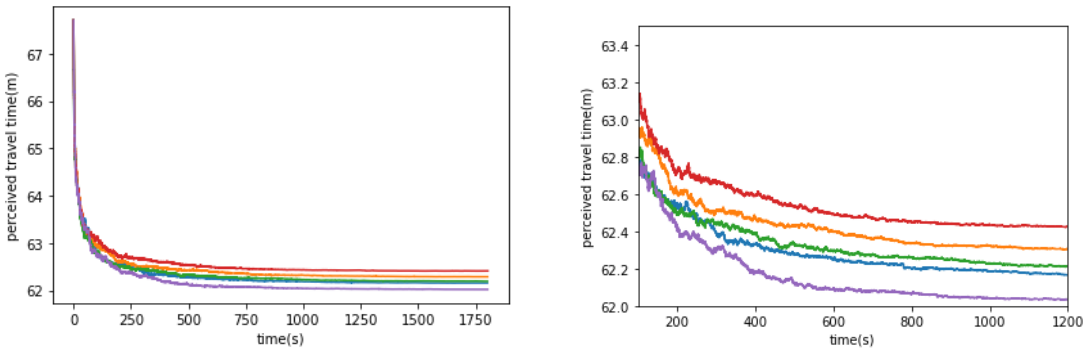
(a) Different settings for the initial temperature temperature.

(b) Different settings for the Decrease of the temperature.

Figure 7.5: Perceived travel time of the best solution found over time for Simulated Annealing executions with different temperature settings.

7.2.3 Order of neighborhoods

Simulated Annealing employs the same shuffled list of neighborhoods as used in Local Search. Figure 7.6 illustrates the results of Simulated Annealing for five different shuffles of all neighborhoods. Initially, all executions demonstrate similar sharp improvements within the first 30 seconds. However, after 1 minute, noticeable differences occur, with perceived travel times of 63.31, 63.18, 63.12, 63.27, and 63.23 minutes. Ultimately, the output timetables from the five Simulated Annealing runs have perceived travel times of 62.16, 62.28, 62.19, 62.41, and 62.02 minutes. This range of 0.39 minutes between the best and worst output highlights the benefit of running Simulated Annealing multiple times with different neighborhood shuffles. Notably, only one execution performed worse than the best result of Local Search, suggesting that Simulated Annealing is likely to outperform Local Search for the given configurations.



(a) Full 20 minutes.

(b) Removed first 2 minutes.

Figure 7.6: Perceived travel time of the best solution found over time for Simulated Annealing executions with different order of the neighborhoods.

For Simulated Annealing, it's important to note that it does not necessarily find a local minimum, as it continues until no improvement is observed for 1000 steps. However, Local Search can be applied after Simulated Annealing to further refine the solution. This results in an improvement of the best found timetable from 62.02 perceived minutes to 61.98 perceived minutes, showing that the timetable was already close to a local minimum. Additionally, Simulated Annealing can be stopped early if a promising timetable is found, followed by Local Search to achieve a local minimum quicker. This combined approach can be beneficial, as Simulated Annealing produces fast improvements, while Local Search can provide more significant enhancements by focusing on neighbourhoods around an already good timetable.

7.3 Observations on best found timetable

In this section, the best found timetable is evaluated for multiple aspects. Firstly, the focus will be on how well the timetable performs compared to the lower bounds. Next, the performance of the preselected routes for the best found timetable is considered. After this, the timetable's performance for the OD-pairs excluded in the reduced OD-matrix is assessed. Finally, the effectiveness of the timetable in serving passengers is analyzed by comparing the perceived travel times of passengers with the best lower bound to measure possible improvements for passengers.

7.3.1 Best found timetable versus lower bounds

In Figure 7.7, the best output of Simulated Annealing is compared to the precalculated lower bounds. These bounds are calculated using the three methods introduced in Section 5.3 : the shortest route per OD-pair (dotted line), the shortest route per departure (dashed line), and the best route-departure combination (solid line). The lower bounds

have respectively a perceived travel time of 56.86, 59.27 and 60.04 minutes. The substantial improvement of the lower bound with 2.41 perceived minutes from shortest route per OD-pair to shortest route per departure indicates that this method is often significantly better than simply using the shortest route per OD-pair. Further, searching for the best route per departure yields additional significant improvement of 0.77 perceived minutes, highlighting the impact of frequency decreases on the lower bound. While the best Simulated Annealing execution shows clear progress towards the best lower bound, the improvement stagnates after the first ten minutes. The final output of the Simulated Annealing execution is 62.02 perceived minutes, which is a difference of 1.98 perceived minutes to the best lower bound. This difference is relatively small, but the difference suggests that the lower bound under performed since it does not consider extra transfer time or that the neighborhoods in the heuristics are too restricted by focusing only on single train services or single directions of a train line.

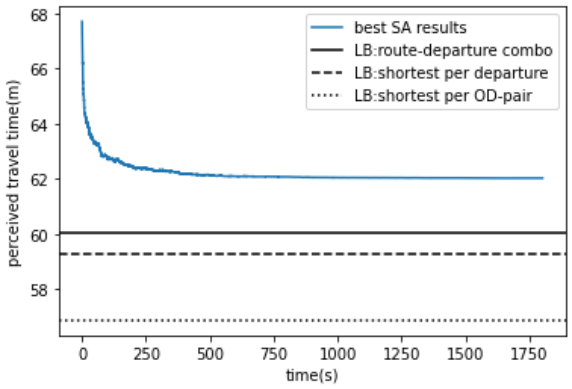


Figure 7.7: Perceived travel time of the best solution found over time for the best found Simulated Annealing output against the three lower bounds(LB): best route-departure combination, shortest per departure of OD-pair and shortest per OD-pair.

7.3.2 Effect of preselected routes and reduced OD-matrix

The impact of the preselected routes and the reduced OD-matrix is given in Table 7.2. This table shows the perceived travel time for the reduced OD-matrix and the complete OD-matrix, including comparisons for preselected routes, all routes with Dijkstra's algorithm, and the calculated best lower bound with departure-route combinations.

For the heuristics, a set of preselected routes is used instead of all possible routes, which can be calculated with Dijkstra's algorithm. The effect of these preselected routes is evaluated by comparing the perceived travel times for the best found timetable using preselected routes against using Dijkstra's algorithm. The difference between the perceived travel times with preselected routes and Dijkstra's algorithm is 0.0084 for the reduced OD-matrix. This difference is negligible compared to the overall perceived travel time,

indicating that the preselected routes adequately represent all possible routes. Furthermore, the difference between the perceived travel times with preselected routes and Dijkstra's algorithm is 0.0112 for the complete OD-matrix. The increase in difference from 0.0084 for the reduced OD-matrix to 0.0112 for the complete OD-matrix exceeds the 10 percent of removed passengers, which would suggest that the difference for the complete OD-matrix should be $0.0084 \cdot \frac{10}{9} = 0.0093$. Since, the perceived travel time is higher with the complete OD-matrix, this means that the removed OD-pairs had a higher perceived travel time. This suggests that the additional deviation for the preselected routes likely means that the preselected routes are evaluated worse for the removed OD-pairs.

The original OD-matrix consists of 5,365 OD-pairs with non-zero passenger demand. The lowest passenger demand for an OD-pair is 0.55, and the highest is 25,295 passengers. The median passenger demand is 53, and the average is 259, indicating that the OD-matrix is composed of many small OD-pairs and a few very large ones. The reduced OD-matrix, created by removing ten percent of the total passenger demand from the least important OD-pairs, consists of 1,642 OD-pairs. The lowest passenger demand in this reduced matrix is 124, with a median of 301 and an average of 762. This shows that while only 10 percent of the passengers are removed, 69 percent of all OD-pairs are excluded.

The effect on the removed OD-pairs is evaluated by observing the difference between the lower bound and the best found solution for the reduced OD-matrix and the complete OD-matrix. The difference between the perceived travel time of the best lower bound and the best output timetable is approximately 1.94 minutes for the reduced OD-matrix and approximately 2.47 minutes for the complete OD-matrix. This indicates that the complete OD-matrix results in a greater discrepancy compared to the 10 percent of removed OD-pairs, which would suggest that the difference for the complete OD-matrix should be $1.94 \cdot \frac{10}{9} = 2.16$. This increased difference is probably because the removed OD-pairs primarily have longer perceived travel times, which requires multiple transfers. Furthermore, the lower bound calculation does not take into account additional transfer times, leading to a larger gap between the objective value and the lower bound for these removed OD-pairs.

Table 7.2: Objective value for the used OD-matrix for a given route set and the minimum over the routes.

	reduced OD-matrix	complete OD-matrix
preselected routes	61.98003	69.81131
Dijkstra's algorithm	61.97161	69.79998
best lower bound	60.03630	67.32268

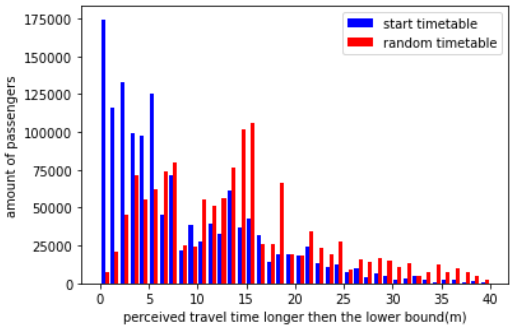
7.3.3 Influence of best found timetable on the passengers

Figure 7.8 presents two histograms illustrating the number of passengers versus the difference between their perceived travel time and the minimum perceived travel time per

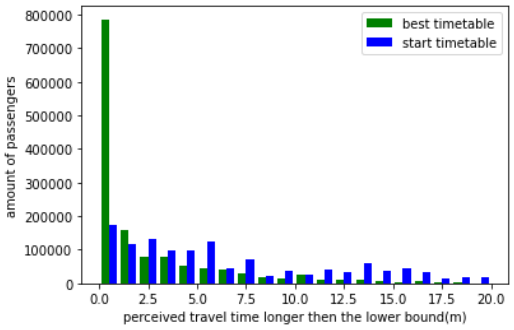
OD-pair derived from the best lower bound. These histograms are based on the passenger volume and average perceived travel time for each OD-pair. By subtracting the minimum average perceived travel time from the average travel time of the timetable for an OD-pair, the improvement potential for passengers is calculated. This information, combined with the passenger demand for each OD-pair, provides a clear view of how effectively passengers are served. The histograms both represent 1,251,625 passengers, categorized into bins of 1-minute.

Figure 7.8a illustrates the difference between a random timetable and the initial timetable used for the heuristics. The random timetable features randomly assigned start times for train services with minimal drive and dwell times. In contrast, the initial timetable is based on this random timetable but with a forced 30-minute difference between train services of the same line in the same direction. The figure reveals that the random timetable results in only a few passengers having a perceived travel time of up to 1 minute longer than the minimum. However, in the initial timetable, one seventh of the passengers experience a perceived travel time up to 1 minute longer than the minimum. The random timetable has an average perceived travel of 16 minutes longer than the minimum, while the start timetable has only an average 8 minutes longer. The histogram for the random timetable is more evenly distributed around the average of 16 minutes, whereas the initial timetable histogram shows a clear decrease as the time difference increases. Additionally, the random timetable has a maximum difference of 110 minutes longer, compared to 66 minutes for the initial timetable, indicating that the random timetable includes more extreme outliers with poorly connected OD-pairs. This comparison shows the significant impact of a 30-minute difference on the perceived travel time for passengers.

Figure 7.8b depicts the difference between the best-found timetable and the initial timetable. Both timetables show that a large amount of passengers experience a perceived travel time of no more than 1 minute longer than the minimum. For the best timetable, this condition applies to approximately two-thirds of all passengers. The distribution of perceived travel times is similar for both timetables, with a noticeable decrease in the number of passengers as the time difference increases. However, this decrease is more pronounced for the best timetable. The best timetable achieves an average difference of 2.47 minutes, representing a significant improvement over the 8 minutes for the initial timetable. Additionally, the maximum difference for the best timetable is 36 minutes longer than the minimum, which is considerably lower than the 55 minutes observed for the initial timetable. This 36-minute difference is probably associated with an OD-pair requiring two transfers, as the lower bound does not account well for transfer times, leading to an extra 18 minutes of waiting per transfer. Despite this, the result remains favorable, particularly for OD-pairs with very low passenger demand.



(a) Start timetable vs random timetable.



(b) Best timetables vs start timetable.

Figure 7.8: Histograms illustrating the possible improvement for passengers for the start, initial and best found timetable.

7.3.4 Improvement on the type of travel

In Table 7.3, data is presented on direct and indirect passengers for both the initial timetable and the best timetable found by the heuristics. "Direct time" represents the part of the total perceived travel time attributed to passengers taking a direct connection, while "indirect time" represents the total perceived travel time for passengers taking an indirect connection. "Direct passengers" denotes the number of passengers taking a direct connection, and "indirect passengers" denotes the number of passengers taking an indirect connection.

After the improvements, there is a slight increase in the number of indirect passengers, likely due to the better transfers created by the heuristics. Overall, the decrease in direct perceived travel time is larger than the decrease in indirect perceived travel time, although this is influenced by the additional indirect passengers. While 64 percent of the passengers are direct passengers, only 42 percent of the perceived travel time is accounted for by direct passengers. This highlights the importance of indirect passengers due to their higher perceived travel times.

Table 7.3: Perceived travel time accounted to direct and indirect passengers with the amount of direct and indirect passengers for the start timetable and the best found timetable.

	direct time	indirect time	direct passengers	indirect passengers
Start timetable	32.93	42.96	895799	495096
Best timetable	29.33	40.52	881035	509860

In Table 7.4, the influence of drive, transfer, and initial waiting time on the total perceived travel time is shown for the initial and the best-found timetable together with the amount of unavoidable pain from the 20 penalty minutes per transfer and the minimum transfer time of 3 minutes. The decrease in drive time from the initial to the best timetable suggests that longer routes are better avoided, as the initial timetable already has the drive

and dwell times set to the shortest possible durations, meaning that these can no longer be made shorter. The decrease of 1.44 perceived minutes in transfer time highlights the significance of optimizing transfer times. This is particularly important because the unavoidable transfer time has increased, indicating that transfers are utilized more frequently in the best-found timetable. The largest improvement is observed in the initial waiting time, indicating the importance of evenly spread departures throughout the hour. Overall it is noticeable that 1.87 of the 2.47 perceived minutes between the best found solution and the best lower bound is due to possibly avoidable transfer time, likely indicating the importance of the missing transfer times in the lower bound.

Table 7.4: Perceived travel time accounted to drive, transfer and initial wait time with the perceived travel time accounted by the unavoidable 23 minutes for a transfer for both the start and best found timetable.

	drive time	transfer time	initial wait time	unavoidable transfer time
Start Timetable	48.88	12.63	14.38	8.81
Best Timetable	47.74	11.19	10.93	9.32

Chapter 8

Findings and conclusions

8.1 Summary

This thesis addresses the problem of Strategic Passenger-Oriented Timetabling (SPOT). SPOT involves assigning departure and arrival times to train services, with a primary focus on optimizing the passenger experience on the train network. SPOT is performed during the early stages of planning, where the feasibility of the timetable with respect to the existing infrastructure is less critical. Instead, SPOT serves as a intermediary between the passenger-focused challenge of line planning and the more feasibility-oriented task of tactical timetabling.

In SPOT, a timetable is evaluated by considering the perceived travel time of passengers, which includes, the actual travel time, penalties for transfers, and penalties for extra initial waiting time at the start station. The perceived travel time is calculated based on the assumption that passengers will always choose the route with the shortest travel time. The objective value of SPOT is then defined as the average perceived travel time of all passengers, where each origin-destination-time-triple (ODT-triple) contribution is weighted by the corresponding passenger demand. "Time" refers to the preferred departure time of a passenger, which is assumed to be evenly distributed throughout the hour. As a result, the focus is on passenger demands for origin-destination-pairs (OD-pairs), which ensures that the timetable minimizes the overall travel experience for passengers across the network.

Creating a timetable in SPOT is based on the principles of the Periodic Event Scheduling Problem (PESP), where an hourly timetable is designed to be repeated over several hours. In PESP formulations, various constraints are applied to ensure that the timetable is feasible, safe, and comfortable for passengers. However, in the context of SPOT, where the feasibility on the existing infrastructural network is less of a concern, safety constraints are de-emphasized and therefore removed. Likewise, comfort constraints are excluded because SPOT aims to optimize the overall passenger experience, thus eliminating the need for comfort-focused constraints. This leaves only the essential constraints that enforce the minimum and maximum times for drive and dwell activities for each train service. As a

result, there are no constraints linking different train services, which significantly simplifies the process of identifying feasible timetables.

In SPOT, the exact objective value of a timetable is determined by employing Dijkstra's shortest path algorithm, which is used to calculate the shortest perceived travel time for passengers traveling from a departure at one station to an arrival at another station. For each passenger, the perceived travel time is minimized by considering both the initial waiting time before departure and the subsequent travel time. The representation of the train network is optimized in several ways to enhance the performance of Dijkstra's algorithm, ensuring that the calculation of these shortest perceived travel times is efficient.

To simplify the process of solving SPOT, several strategies are implemented. The first strategy involves grouping passengers who are certain to use the same route, specifically those with preferred departure times between two consecutive train departures at a station. Another strategy is to use a preselected set of routes for calculating perceived travel times rather than relying on Dijkstra's shortest path algorithm. This method is more efficient when the preselected set is small, although some routes can be excluded if the probability of using them is minimal. As long as this probability remains low, the preselected routes can still accurately represent the complete set of possible routes. The final strategy involves excluding OD-pairs with lower passenger demands from the calculations. This further speeds up the process by reducing the number of routes that need to be considered. Since many OD-pairs are rarely used but require the same computational effort as those with higher demand, focusing on more frequently used OD-pairs significantly enhances the efficiency of the solution process.

SPOT was initially implemented using a Mixed-Integer Linear Programming (MILP) formulation, but this approach faced three major challenges. First, the MILP was only fast for small instances with a limited number of train services, making it unsuitable for larger, real-world scenarios. Second, the process of generating an initial timetable using the MILP was time-consuming, even though the creation of an initial timetable should be relatively straightforward. Finally, the gap between the lower bounds identified by the MILP and the best-found timetable was relatively large, making it difficult to evaluate whether the timetable was performing well. Consequently, the MILP formulation did not provide a reliable measure of the timetable's effectiveness.

In this thesis, a heuristic approach is employed to address strategic passenger-oriented timetabling, avoiding the issues associated with the MILP formulation. Two heuristics are utilized: Local Search and Simulated Annealing.

For Local Search, neighborhoods are defined around a timetable. Within such a neighborhood, all possible timetables are evaluated, and the best timetable becomes the new current timetable. This process is repeated over multiple neighborhoods to iteratively improve the timetable. For this the chosen neighborhood follows a predefined list of neighborhoods,

which defines the order in which the neighborhoods are checked. This list is repeated until a local minimum is found.

Simulated Annealing, on the other hand, also explores the same list of neighborhoods but introduces randomness by selecting a random timetable within the neighborhood. The probability of accepting a timetable depends on the perceived travel time of the timetable compared to the current timetable. If the new timetable is better, then it is always accepted and if it is worse, then the chance decreases the worse the timetable is. A temperature parameter is introduced to control the likelihood of accepting worse timetables, with higher temperatures increasing the chances of acceptance. This temperature gradually decreases, balancing between improving the timetable and maintaining the possibility of escaping local minima.

For the neighborhoods, multiple types are defined, including shifting entire train services, shifting dwell times and shifting drive times. While additional drive times and additional dwell times are nearly equivalent in their effects, extra dwell times generally offer better results. Consequently, the focus narrows down to train service shifts and dwell time shifts. Train service shifts significantly benefit direct passengers by optimizing the scheduling of the direct connections, while keeping the in-train time low. On the other hand, dwell time shifts primarily enhance the experience of indirect passengers by increasing the in-train time for better transfer times.

For SPOT, multiple lower bounds are defined to evaluate timetable performance. The first lower bound is based on the shortest possible route between stations, considering only direct travel times without additional factors. This bound is then refined by incorporating the initial waiting time, assuming evenly spread departures while still using the shortest possible route. The third lower bound further improves accuracy by considering the shortest route for each specific departure, with an optimal spread of departures calculated to minimize perceived travel time. Finally, the most advanced lower bound accounts for frequency decreases by evaluating routes that utilize the same train service after a transfer. If two routes use the same train service following the same number of transfers, one route will always be more efficient, leading to an improved lower bound by considering all possible route-departure combinations.

For both Local Search and Simulated Annealing, optimal heuristic settings were explored to improve the quality of the generated timetables. It was observed that dwell time shifts play a significant role in enhancing the heuristics by facilitating better transfer options. In the case of Local Search, the potential for smaller neighborhoods to increase computational speed was examined. However, the results indicated that the speed gains were negligible and came at the expense of worse local minima. For Simulated Annealing, optimal settings for the initial temperature and its decay were determined, balancing the need to escape local minima with the goal of gradually improving the best-found timetable. Additionally, the effect of altering the order in which neighborhoods were explored was

assessed for both Local Search and Simulated Annealing. The findings revealed significant differences in outcomes depending on the neighborhood order, indicating that running multiple iterations with varied neighborhood sequences can be beneficial for both heuristics.

Finally, the best-found timetable generated by the heuristics was analyzed, and the effects of key decisions were evaluated. First, the impact of using preselected routes and a reduced OD-matrix as approximations was assessed. This evaluation focused on how these choices influenced the accuracy and efficiency of the solution. Second, the improvement for passengers was examined by comparing the best-found timetable against the potential improvements suggested by the calculated lower bounds. Finally, the timetable's impact on direct versus indirect passengers was analyzed, with a detailed breakdown of how drive time, transfer time, and initial waiting time contributed to the total perceived travel time for passengers.

8.2 Conclusions

The primary focus of this thesis was to use heuristics to address challenges associated with the MILP formulation for SPOT. In terms of scalability, both Local Search and Simulated Annealing perform better than the MILP formulation for larger problem cases, such as the intercity network of NS. The heuristics show significantly fast improvements with Simulated Annealing producing good timetables even within minutes. Additionally, the introduction of the lower bounds provides a clearer understanding of how close a found timetable is to the optimal solution. These lower bounds, evaluated per OD-pair, not only offer insights into the potential improvements for the current timetable but also help in identifying specific areas where enhancements can be made.

Overall, Simulated Annealing shows a slight advantage over Local Search with the current settings. Not only does Simulated Annealing produce good results about ten times faster, but it also yields slightly better timetables. Almost all timetables generated by Simulated Annealing perform better than those produced by Local Search. Furthermore, the variation in results between different neighborhoods orders in Simulated Annealing indicates that running the algorithm multiple times with different neighborhood orders can lead to significant improvements.

The heuristics achieved a clear improvement for both direct and indirect passengers. It is striking that the best timetable shows a significant improvement in transfer efficiency: more passengers are traveling indirectly compared to the initial timetable. This improvement is reflected in the relatively larger increase in unavoidable transfer time, suggesting that routes with multiple transfers have become more effective. Potential avoidable transfer times were almost halved, indicating that significant progress has been made in reducing unnecessary transfers. In addition the initial waiting time has been significantly improved, which underlines the importance of an optimal spread of departure times. The improvement in drive time was minimal, suggesting that the drive times were already close to optimal.

Almost the entire difference between the objective value of the lower bound and the best-found timetable is attributed to potentially avoidable transfer time. This suggests that the lower bound can still be improved, indicating that the timetables are likely much closer to optimality than initially assumed.

Overall, the preselected routes and the reduced OD matrix represent the complete problem relatively well. The preselected routes led to only a minor difference in the objective value compared to considering all possible routes. The removed OD-pairs from the reduced OD matrix resulted in a slightly larger difference from the lower bounds, but this is likely due to longer routes with multiple transfers for the removed OD-pairs.

8.3 Further research

The main issue with the current heuristics is that they adjust only one train service at a time, potentially missing larger, more beneficial changes. While expanding the size of these neighborhoods could yield more improvements, it also significantly increases the computation time. To address this, it would be more effective to focus on how well passengers are served in the current timetable and to target timetables with a greater chance of improvement. This approach would involve exploring shifts that affect multiple train services simultaneously, although identifying such opportunities can be challenging.

Another option is to develop improved lower bounds that better account for additional transfer time. This research is worthwhile because the current lower bound, while performing well for direct passengers, often falls short for indirect passengers. Since much of the difference between the lower bound and the best-found timetable is attributed to extra transfer time, it is crucial to understand how this missing transfer time affects performance and to identify potential improvements for long-distance routes. By enhancing the lower bound, it becomes possible to more accurately assess where the opportunities for improvement lie and to make better informed decisions about optimizing the timetable.

All calculations in this thesis were performed using Python, an interpreted language in which code is translated into machine code during execution. This approach leaves room for significant performance improvements. Transitioning the algorithms to a compiled language like C could potentially enhance performance by at least a factor of ten. Additionally, Python's Global Interpreter Lock (GIL) limits the effectiveness of multi-threading by ensuring that only one thread executes Python code at any time, despite the presence of a multi-threading library. To fully utilize multi-threading and further accelerate computations, implementing the algorithms in a language that supports true parallel execution would be beneficial.

Finally, a more detailed investigation into the effects of preselected routes and the reduced OD-matrix would be beneficial. Although the current findings suggest that the chosen routes are effective, given the minimal difference compared to using all routes, it

remains valuable to examine whether further reductions in the set of preselected routes are feasible without compromising performance. Furthermore, for the removed OD-pairs, it could be useful to explore whether retaining smaller OD-pairs might enhance the focus on transfers, potentially leading to further improvements in the timetable.

Bibliography

[1] Simon Henry Bull, Richard Martin Lusby, and Jesper Larsen. An optimization based method for line planning to minimize travel time. In *13th Conference on Advanced Systems in Public Transport*. Erasmus University, 2015.

[2] Sofie Burggraeve, Simon Henry Bull, Pieter Vansteenwegen, and Richard Martin Lusby. Integrating robust timetabling in line plan optimization for railway systems. *Transportation Research Part C: Emerging Technologies*, 77:134–160, 2017.

[3] Michael Bussieck. *Optimal lines in public rail transport*. PhD thesis, Techinical University of Braunschweig, 1998.

[4] Gabrio Caimi, Leo Kroon, and Christian Liebchen. Models for railway timetable optimization: Applicability and applications in practice. *Journal of Rail Transport Planning & Management*, 6(4):285–312, 2017.

[5] Bart de Keizer, Marco Kouwenhoven, and Freek Hofker. New insights in resistance to interchange. *Transportation Research Procedia*, 8:72–79, 2015.

[6] Steven S Harrod. A tutorial on fundamental model structures for railway timetable optimization. *Surveys in Operations Research and Management Science*, 17(2):85–96, 2012.

[7] Gert-Jaap Polinder. *New Models and Applications for Railway Timetabling*. PhD thesis, Erasmus University Rotterdam, 2020.

[8] Giorgio Sartor, Carlo Mannino, Thomas Nygreen, and Lukas Bach. A MILP model for quasi-periodic strategic train timetabling. *Omega*, 116:102798, 2023.

[9] Paolo Serafini and Walter Ukovich. A mathematical model for periodic scheduling problems. *SIAM Journal on Discrete Mathematics*, 2(4):550–581, 1989.

[10] Nederlandse Spoorwegen. Ns jaarverslag 2023. `https://www.nsjaarverslag.nl`, 2024.

[11] Fei Yan and Rob MP Goverde. Combined line planning and train timetabling for strongly heterogeneous railway lines with direct connections. *Transportation Research Part B: Methodological*, 127:20–46, 2019.

[12] Chuntian Zhang, Jianguo Qi, Yuan Gao, Lixing Yang, Ziyou Gao, and Fanting Meng. Integrated optimization of line planning and train timetabling in railway corridors with passengers' expected departure time interval. *Computers & Industrial Engineering*, 162:107680, 2021.