ERASMUS UNIVERSITY ROTTERDAM

ERASMUS SCHOOL OF ECONOMICS

Master's thesis

Econometrics and Management Science

Operations Research and Quantitative Logistics

Line Planning with Transfer Decisions

Author

Y.K. Cheung (377511)

Supervisor

dr. T.A.B. Dollevoet

Second assessor

B.T.C. van Rossum

July 31, 2022

Abstract

This thesis focuses on the line planning problem with transfer decisions. We use a mathematical formulation of the line planning problem and extend that to incorporate transfer decisions. By extending the formulation, we ensure that these transfer decisions are provided and are valid in cyclic timetables that are generated in the next planning phase of public transport planning. We propose a constraint generation algorithm in order to solve the problem, as the number of constraints can grow exponentially by the instance size. The proposed algorithm is applied to artificial instances and a modified real-world instance. Results show that the line planning problem with transfer decisions can provide useful transfer decisions that can be considered in the timetabling phase of public transport planning.

Contents

List of Symbols

$\mathcal{S}$	set of stations in PTN
E	set of edges in PTN
V	set of vertices in the passenger flow model
A	set of arcs in the passenger flow model
$\mathcal{E}$	set of events in EAN
$\mathcal{A}$	set of activities in EAN

Chapter 1

Introduction

One of the most important factors of the attractiveness of public transport is convenience (Warman, 2014). The convenience of public transportation depends on access and egress time, waiting time, expected delay time, mean crowding, number of transfers, and access to information (Warman, 2014). Service with a high frequency leads to lower waiting times and transfer times.

One of the influencing factors for convenience might be whether there is a direct connection from one's departure location to one's arrival location. One can imagine that a direct service is more attractive than a service in which a passenger has a transfer at an intermediate location. In public transport, however, direct connections can not always be offered. Hence, short transfer times are also important for the experience of passengers. To actually determine the transfer times, a timetable is required. However, we want to incorporate the transfer time in the line planning problem as well.

In this thesis, we focus on incorporating transfer decisions into the line planning problem. Line planning is finding a set of lines and corresponding frequencies. Given the infrastructure of a public transport system and the passenger flows in the network, the aim is to estimate the total traveling time of the passengers more accurately by considering whether a short transfer can be offered by the line plan or not. These short transfers can be of use in the timetable part of the planning process in public transport. Moreover, these short transfers allow for better approximated transfer times in the line planning phase. The objective is to minimize the total travel time for the entire public transport network, subjected to the limitations in the network and the limited short transfers that can be guaranteed in a network.

Current line planning problems approximate the transfer times, which measures the total travel time of the passengers incorrectly. For instance, a passenger on a line with a high frequency (e.g. 4 times an hour) needs to change to a line that is only offered once an hour. The decision to offer a quick transfer depends on whether the transfer is important enough in the whole public transport network.

1.1 Planning in Public Transport

The planning process in public transport is complex and therefore divided into several consecutive phases. These phases are on *strategic*, *tactical* and *operational* level and are further subdivided into several subproblems. Huisman et al. (2005) provide an overview of the planning problems that arise at Netherlands Railways and we roughly base this section on this overview.

Network design and line planning are part of strategic planning and their planning horizon is several years. The strategic planning is based on an origin-destination (OD) matrix, which consists of passenger demand for each origin to each destination in the network. The objective of the *network design problem* is to minimize the infrastructure costs by choosing the optimal sets of links in the network that are needed to operate public transport. Subsequently, the *line planning problem* (LPP) aims to determine the lines and frequencies in a public transport network, such that all travel demand is satisfied. There are two main objectives in the line planning problem, either the service for the passenger is maximized or the operational costs of the public transport system are minimized.

The tactical planning consists of timetabling using the solution of lines and frequencies from the line planning problem. The planning horizon is seasonal, which can be every three months up to once a year. The aim is to find a timetable, which contains service trips with departure and arrival locations and times. From the timetable, the service trips can be created for the next phase. Moreover, a timetable allows us to compute the travel times for the passengers.

Vehicle scheduling and crew scheduling are part of the operational planning and their planning horizon is usually a day. The *vehicle scheduling problem* aims to allocate the service trips that need to be executed by the vehicles while minimizing the operational costs. The problem also includes deadhead trips, which are trips without any passengers, e.g. a trip between two passenger trips or a trip from and to a shunting yard. From the solution to the vehicle scheduling problem, crew tasks can be created. A crew task is a piece of work between two relief points, a point where a crew member can change vehicles. The objective of the *crew scheduling problem* minimizes the costs of the crew that are assigned to the vehicle trips. These two scheduling problems can be integrated in order to reduce the total costs of vehicle and crew scheduling.

In Figure 1.1, the planning process of public transport is shown as a schematic overview. This figure is largely based on the figure in Liebchen (2008).

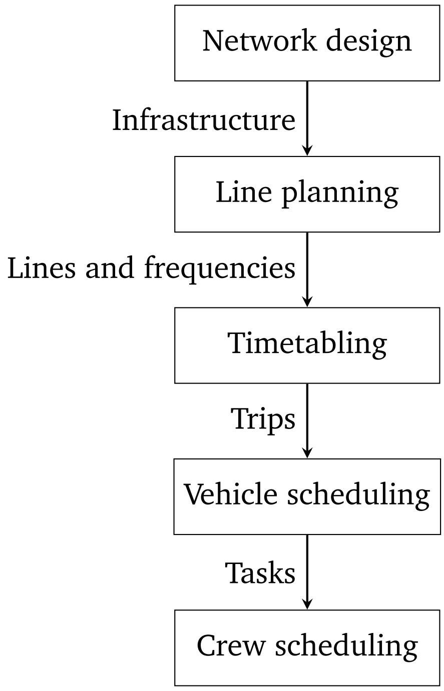

Figure 1.1: Schematic overview of the planning process in public transport (Liebchen, 2008).

1.2 Outline of this thesis

The remainder of this thesis is as follows. A discussion of the current literature can be found in Chapter 2. Chapter 3 provides the mathematical formulation for the line planning problem. Chapter 4 focuses on timetabling and provides an addition to the mathematical formulation that incorporates the transfer decisions. Chapter 5 elaborates on the solution approach to the line planning with transfer decisions. In Chapter 6, the computational results of the proposed method are presented. Finally, the conclusion of this thesis is provided in Chapter 7.

Chapter 2

Literature Review

In this chapter, we discuss the literature on line planning and timetabling in public transport. There is a substantial amount of research available on line planning and timetabling. A selection of the relevant literature is treated in the following sections of this chapter.

2.1 Line Planning

In line planning, one can distinguish between cost-oriented models and passenger-oriented models. The former focuses on finding the lowest operational costs, while the latter focuses more on for example the travel time of the passengers in the public transport system.

2.1.1 Cost-oriented models

Cost-oriented models focus on finding the lowest operational costs with respect to passenger demand. Claessens et al. (1998) presented an extensive research on cost-oriented models for the LPP. The goal of their work is to find lines subject to service constraints and capacity requirements. The authors present an integer non-linear programming formulation and transform this formulation into a linear one, in order to solve it using branch-and-bound. The model presented by Claessens et al. (1998) also determines the vehicle type operating the line and the train length.

The research of Goossens et al. (2004) approaches the LPP with branch-and-cut using a similar model used by Claessens et al. (1998). Their main contribution is an extensive preprocessing process and the development of valid inequalities for this problem in order to tighten the lower bound. Goossens et al. (2006) extend their approach to a *multi-line* *planning problem* in which the line system could have multiple line types with different stopping patterns.

2.1.2 Passenger-oriented models

In passenger-oriented models, the assumption is that the passengers choose their optimal route in terms of travel time. The focus is not primarily on costs, although cost constraints are usually included.

Schöbel and Scholl (2006) present an approach to minimize the total travel times for passengers including penalizing the transfers. In order to incorporate this, the authors present a change-and-go network that replaces the infrastructure network as the underlying network for the mathematical formulation. The change-and-go network combines stations and lines as one vertex in a graph and thus integrates line planning and traffic assignment. The authors solve the problem with LP relaxation using Danzig-Wolfe decomposition. The disadvantage is that this leads to long solution times due to large IP models.

Borndörfer and Neumann (2010), Borndörfer et al. (2007), and Borndörfer and Karbstein (2012) present a model in which passengers can be freely routed and use column generation to generate those routes. In their research, the objective is to minimize the riding time and therefore they neglect the transfer times. The authors assume that the transfer time is independent of the line frequency.

In the models introduced by Goerigk and Schmidt (2017) and Schmidt (2014), only line concepts that allow all passengers to travel on the shortest path are considered feasible. The authors propose an IP formulation and present a genetic algorithm to solve the problem.

Bull et al. (2019) present a model that optimizes the total travel time and incorporates frequency-dependent transfer costs and integrates passenger routing into the line planning problem. The authors analyzed their performance on instances taken from a commuter train network in Denmark.

2.2 Timetabling

In timetabling, one can distinguish between cyclic and non-cyclic timetabling. A cyclic timetable has the property that an event, a departure or an arrival, occurs in a periodic way. For example, train service at an hourly frequency departs 10 minutes past every hour, so at 9:10, 10:10, etc. This makes it easier for passengers to remember the timetable. The drawback is that this timetable is more expensive to operate. Non-cyclic timetabling does not have this property. In this thesis, we focus on cyclic timetabling.

Most cyclic timetabling is based on the periodic event scheduling problem (PESP) by Serafini and Ukovich (1989). PESP aims to find a feasible schedule at which the periodic recurring events take place. This problem introduced by Serafini and Ukovich (1989) is a feasibility problem and does not have an optimization objective. Nachtigall (1996) extends the PESP by adding an objective, namely minimizing the waiting times for the passengers. Moreover, the author transforms the formulation into one in terms of cycles and this is called the cycle periodicity formulation. Odijk (1996) uses a constraint generation algorithm in order to construct periodic railways timetables at stations. Nachtigall and Voget (1996) use PESP to obtain a timetable with minimum waiting times and use a genetic algorithm to obtain a timetable.

In the PhD thesis of Peeters (2003), the author applied the timetabling problem to the Dutch railway network. The author also extended the formulation with variable trip times. Furthermore, the author provided optimization of timetabling with other objectives, for example, maximizing the timetable robustness and minimizing the required number of rolling stock compositions.

Caprara et al. (2007) and Cacchiani and Toth (2012) offer an elaborate survey on all railway timetable optimization.

2.3 Integrated approaches

In the last two decades, there is more focus on integrating several steps of the planning process (as shown in Figure 1.1). The optimization of the line plan with a corresponding timetable and vehicle schedule can be seen as a multi-stage optimization problem (P. Schiewe, 2020). The rationale behind the integration is that optimizing problems sequentially leads to suboptimal results and integration can lead to optimal solutions. Schöbel (2017) presents an *eigenmodel* for the whole public transportation optimization. In the presented approach, the author integrates line planning, timetabling, and vehicle scheduling in a bi-objective model. That is, the output of timetabling or vehicle scheduling can serve as the input of the line planning phase. Consequently, the line planning problem can be reoptimized with respect to the limitations in timetabling or vehicle scheduling.

Chapter 3

Line Planning

In this chapter, we first define the basic elements of line planning. Then, we provide the model formulation for the line planning problem.

3.1 Basic definitions

A public transport network can be defined as follows.

Definition 1 (PTN). A *Public Transport Network* $PTN = (\mathcal{S}, E)$ is an undirected graph with the stations as the set of vertices $\mathcal{S}$ and the connections between them as the set of edges $E \subset \mathcal{S} \times \mathcal{S}$.

We define a line l within the PTN as follows.

Definition 2. A *line* l is a path within the PTN represented as a sequence of alternating stations and edges:

$$(s_1, e_{1,2}, s_2, \ldots, e_{k-1,k}, s_k) \tag{3.1}$$

where stations $s_i \in \mathcal{S}$ and edges $e_{i,j} \in E$.

Every line l is operated with a frequency f_l, which denotes how often the service of the line is offered within a certain period, usually an hour. A station can be visited by more than one line. If that is the case, we call this station a *transfer station*.

The combination of the set of lines $\mathcal{L}$ and frequencies for each line define a line concept:

Definition 3. A *line concept* $(\mathcal{L}, f)$ is the set of lines $\mathcal{L}$ that are operated and their frequencies f_l for all $l \in \mathcal{L}$.

The *origin-destination matrix* consists of all OD pairs with demand between the origin station and the destination station and the set of OD pairs is denoted by $\mathcal{P}$. The demand is usually expressed in the number of passengers. A *direct traveler* is a passenger that does not need to change lines in order to get from the origin u to the destination v, where $u, v \in \mathcal{S}$. The *riding time* is defined as the time a passenger is traveling in a vehicle between the origin and the destination, the transfer time is neglected. The *traveling time* is defined as the total time a passenger is traveling, that is, the riding time and the transfer time.

3.2 Model formulation

The model consists of a lines model part, which covers the selection of the lines, and a passenger flow model part, which covers the flow of the passengers.

3.2.1 Lines model

Let $\mathcal{S}$ be the set of stations and each $s \in \mathcal{S}$ be a possible stop for a line l. Every station also has passenger demand from an origin station o and the destination station d, which is denoted by w_{od}. Then, the set $\mathcal{P}$ consists of the non-zero demand for all OD pairs in the public transport system, that is, $\mathcal{P} = \{(o, d), o \in \mathcal{S}, d \in \mathcal{S}, w_{od} > 0\}$. The set $\mathcal{L}$ consists of all lines that are possible within the public transport system. Each line l has a set of frequencies $\mathcal{F}_l \subset \mathbb{N}$, which consists of predefined frequencies.

For every line $l \in \mathcal{L}$ and every frequency $f \in \mathcal{F}_l$, we have the decision variable

$$x_{lf} = \begin{cases} 1 & \text{if line } l \text{ is selected with frequency } f, \\ 0 & \text{otherwise.} \end{cases} \tag{3.2}$$

In practice, there are always limitations in the public transport network or requirements required by the public transport authority, such as a maximum budget. Every line has a cost c_{lf} which is frequency dependent and there is a budget c_{max}. Now, the constraints of the lines model can be formulated as follows:

$$\sum_{f \in \mathcal{F}_l} x_{lf} \leq 1 \qquad \forall l \in \mathcal{L}, \tag{3.3}$$

$$\sum_{l \in \mathcal{L}} \sum_{f \in \mathcal{F}_l} c_{lf} x_{lf} \leq c_{\max,} \tag{3.4}$$

$$x_{lf} \in \{0,1\}, \qquad \forall l \in \mathcal{L}, \forall f \in \mathcal{F}_l. \tag{3.5}$$

Constraints (3.3) ensure that for each line at most one frequency is chosen. Constraint (3.4) ensures that the total cost of the lines does not exceed the given budget. Constraints (3.5) ensure that the binary variables take either zero or one.

3.2.2 Passenger flow model

Our model for the passenger flow is based on the model used by Bull et al. (2019) and is partly based on the change-and-go graph created by Schöbel and Scholl (2006). The passenger flow is modeled on a directed graph. As the graph becomes very large, if each line-frequency pair has to be included (Bull et al., 2019), the authors suggest aggregating the line-frequency pairs. The aggregation of the line-frequency pairs is done to reduce the size of the graph.

Let $G = (V, A)$ be the directed graph for the passenger flow part, then the set V consists of the following types of vertices:

- source s^i_{in} and sink s^i_{out} vertex for every station $i \in \mathcal{S}$,
- platform p^i vertex for every station $i \in \mathcal{S}$,
- a station-line s^i_l vertex for every station $i \in \mathcal{S}$ a line l visits, for every $l \in \mathcal{L}$.

Note that the set of vertices V differs from the set of vertices $\mathcal{S}$ defined in the undirected graph PTN in the lines model.

Let A denote the set of arcs as part of the directed graph. Then, $A^l \subset A$ is the subset of all arcs that are part of the line and undetermined frequency, and $A^l_f \subset A$ is the subset of all arcs that have a determined frequency and are used for the long transfers. For the short transfers, we have the subset $A^t \subset A$. The directed graph G contains the following types of arcs. The arcs have a weight that represents the time if an arc is used by a passenger. The arcs in graph G with a duration weight (either riding or transfer time) are the following:

- a travel arc between every adjacent pair of station-line vertices for every line and in both directions with a riding time (these arcs are in A^l),
- an arc from every station platform vertex to every station-line vertex at every frequency for long transfers with a long transfer time (these arcs are in A^l_f),
- an arc from every station-line vertex to every station-line vertex for short transfers with a short travel time (these arcs are in A^t).

Note that, for the long transfers we have parallel arcs between the platform vertex and the station-line vertex depending on the frequency. The weight, therefore, differs at every frequency. The next chapter shows the necessity of the need for extra constraints if we want to incorporate short transfers into the line planning problem.

Then, we have additional arcs that ensure the flows from the origin to the destination, but have a weight of 0. Therefore, the graph G also contains the following arcs:

- an arc from every station-line vertex to every station sink vertex,
- an arc from every station-line vertex to every station platform vertex,
- an arc from every station source vertex to every station-line vertex.

In Figure 3.1, an example of a public transport network is shown. This example has two lines, line l_1 from station W to station E via C and line l_2 from station N to station S via station C. Station C is a station at which a transfer is possible. In Figure 3.2, a directed graph G is (partly) shown for the example of Figure 3.1. Only stations N (green), C (gray), and W (red) are shown, the other stations are omitted in this example. In this example, every station has a source vertex s^i_{in} and a sink vertex s^i_{out} and arcs to the station-line vertex and from the station-line vertex, respectively. The changing station C has two source arcs and two sink arcs as this station is served by both lines. In Figure 3.2, there are two different types of arcs, solid and dotted arcs. The dotted arcs are supporting arcs ensuring the passenger flow from the origin to destination and the traveling time on these arcs is 0. The solid arcs represent a traveling time value and we distinguish three types of arcs with a traveling time: arcs with a riding time, arcs with a short transfer time, and arcs with a long transfer time. The short transfer time is set to t_{short} minutes, the long transfer time is defined as $t_{\text{long}} = \dfrac{60}{f}$, where f is the frequency per hour of the line to which the passenger transfers.

With this directed graph, we can formulate the passenger routing part of the line planning problem as a *multi-commodity flow problem*, where a commodity represents the group

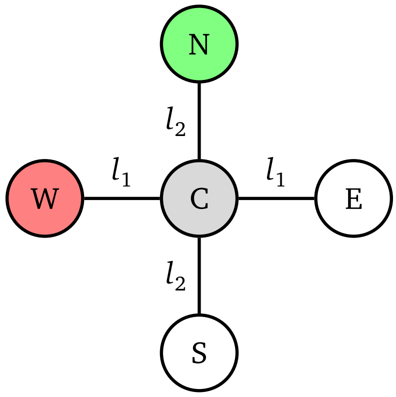

Figure 3.1: An example of a public transport network with the set of stations $\mathcal{S} = \{N, E, S, W, C\}$. At station C both line l_1 and line l_2 have a stop. Line l_1 visits station W, C, and E, and line l_2 visits station N, C, and S.

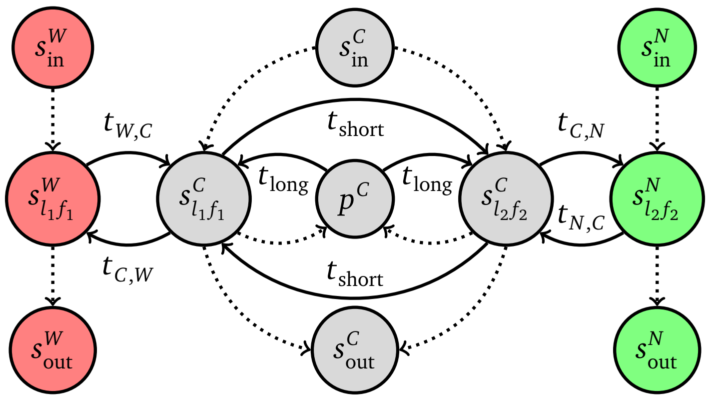

Figure 3.2: An example of a station with two lines. The dotted arcs have a value of 0. The solid arcs represent the travel time. In this example, both lines have just a frequency of one train an hour. If, for example, line 2 has the option to have a frequency of 1 or 2, then we have two arcs with two values of t_{long}, for each of the frequencies.

of passengers with the same origin. The model in Bull et al. (2019) combines flows that have the same origin. This reduces the number of flow decisions by a factor of $|\mathcal{P}|$.

The number of passengers from the origin station o that traverse the arc a is denoted by the flow variables $y_o^a \geq 0$.

Let δ_v^s be the demand for passengers at vertex $v \in V$ whose origin is station s. Then the value of δ_v^s takes the following values:

$$\delta_v^{s_1} = \begin{cases} w_{s_1 s_2} & \text{if vertex } v \text{ is a sink vertex for station } s_2, \\ -1 \cdot \sum_{s_2} w_{s_1 s_2} & \text{if vertex } v \text{ is the source vertex for station } s_1, \\ 0 & \text{otherwise,} \end{cases} \tag{3.6}$$

where $w_{s_1 s_2}$ is the demand from station s_1 to station s_2. The following constraints are then imposed:

$$\sum_{(u,v)\in A} y_s^{(u,v)} - \sum_{(v,w)\in A} y_s^{(v,w)} = \delta_v^s, \qquad \forall s \in \mathcal{S}, \forall v \in V, \tag{3.7}$$

$$\sum_{o\in\mathcal{S}} y_o^a \leq \sum_{f\in\mathcal{F}_l} P_f x_{lf}, \qquad \forall l \in \mathcal{L}, \forall a \in A^l, \tag{3.8}$$

$$\sum_{o\in\mathcal{S}} y_o^a \leq P_f x_{lf}, \qquad \forall l \in \mathcal{L}, \forall f \in \mathcal{F}_l, \forall a \in A_f^l, \tag{3.9}$$

$$0 \leq y_o^a \leq \sum_{s\in\mathcal{S}} w_{os}, \qquad \forall o \in \mathcal{S}, \forall a \in A. \tag{3.10}$$

Constraints (3.7) are flow conservation constraints. Constraints (3.8) ensure that every arc has sufficient capacity for the flows using that arc. P_f denotes the capacity of the line at frequency f and functions as a big-M. Constraints (3.9) ensure that the frequency-dependent long transfer arcs are only used if the line is operated. Constraints (3.10) ensure that the decision variables are non-negative and that the variable is not larger than the sum of all demand from the origin station. The upper bound for this decision variable is set to the maximum demand to obtain tighter bounds for the LP relaxation of the branch-and-bound algorithm.

3.2.3 Objective function

The objective function is to minimize the total traveling time including the transfer time at the station and is as follows:

$$\min \sum_{a \in A} \sum_{o \in \mathcal{S}} t_a y_o^a, \tag{3.11}$$

where t_a is the cost of the arc a and represents the traveling time of a passenger on that particular arc.

Chapter 4

Timetabling

In Chapter 3, we mainly focused on the line planning phase of public transport and we have not yet considered timetabling, which is the next step in the planning process (as shown in Figure 1.1). First, we define the Periodic Event Scheduling Problem, which is generally used in the timetabling phase for periodic timetabling. Afterward, we focus on the cyclic properties in timetables.

4.1 Periodic Event Scheduling Problem

The Periodic Event Scheduling Problem (PESP) is a problem introduced by Serafini and Ukovich (1989) in which events are scheduled and occur in a recurring pattern under periodic time window constraints. The problem can be formulated as follows:

Definition 4 (PESP)**.** The PESP aims to find a periodic schedule given a set $\mathcal{E}$ of events, a set of activities $\mathcal{A} \subseteq \mathcal{E} \times \mathcal{E}$, a cycle time T, and time windows $[l_{ij}, u_{ij}]$ for all activities $(i, j) \in \mathcal{A}$. A periodic schedule $v_i \in [0, T)$ with $i \in \mathcal{E}$ satisfies

$$(v_j - v_i) \quad \text{modulo} \quad T \in [l_{ij}, u_{ij}] \quad \forall (i, j) \in \mathcal{A}. \tag{4.1}$$

Here, v_i and v_j with $i, j \in \mathcal{E}$ represent the time of event i and j respectively. The difference of v_j and v_i is bound by l_{ij} and u_{ij}, these values are respectively the lower and the upper bound of the process time between the events i and j. The cyclicity of the timetable is modeled by the modulo operator. In optimization, the modulo operator in Equation (4.1) is replaced by a binary variable $p_{ij} \in \{0, 1\}$ for all $(i, j) \in \mathcal{A}$, in order to obtain a constraint that is easier to model. That is,

$$v_j - v_i + Tp_{ij} \in [l_{ij}, u_{ij}] \qquad \forall (i,j) \in \mathcal{A}. \tag{4.2}$$

So if we have an activity with event times $v_i = 56$ and $v_j = 4$, and the cycle time $T = 60$. Then, $p_{ij} = 1$ as $v_j < v_i$, so filling in the equation: $4 - 56 + 60 = 8$ minutes. That is how long the activity from event i to j lasts.

To summarize, the PESP formulation is as follows:

$$\begin{aligned}
(PESP) \qquad \min \quad & F(v,p) && && (4.3)\\
\text{s.t.} \quad & l_{ij} \leq v_j - v_i + Tp_{ij} \leq u_{ij} && \forall (i,j) \in \mathcal{A}, && (4.4)\\
& v_i \in [0, T) && \forall i \in \mathcal{E}, && (4.5)\\
& p_{ij} \in \{0,1\} && \forall (i,j) \in \mathcal{A}. && (4.6)
\end{aligned}$$

A PESP instance can also be (graphically) represented by an event-activity network $\mathcal{N} = (\mathcal{E}, \mathcal{A})$. This network $\mathcal{N}$ is a graph with a set of vertices $\mathcal{E}$ representing events and a set of arcs $\mathcal{A}$ representing activities. In our case, the set of events can be divided into two types:

- an arrival event from line l and at station s is denoted by the arrival vertex $a_l^s \in \mathcal{E}_{\text{arr}}$, where $\mathcal{E}_{\text{arr}} \subset \mathcal{E}$,
- a departure event from line l from station s is denoted by the departure vertex $d_l^s \in \mathcal{E}_{\text{dep}}$, where $\mathcal{E}_{\text{dep}} \subset \mathcal{E}$.

Then, there are activities that link two events to each other. We can divide these activities into three types:

- a driving activity $\left(d_l^{s_1}, a_l^{s_2}\right) \in \mathcal{A}_{\text{drive}} \subset \mathcal{A}$ that links the departure event $d_l^{s_1}$ of a line from station s_1 to an arrival event $a_l^{s_2}$ of the same line at the following station s_2,
- a dwelling activity $\left(a_l^s, d_l^s\right) \in \mathcal{A}_{\text{dwell}} \subset \mathcal{A}$ from a line that links the arrival event a_l^s of a line at a station to the departure event d_l^s of the same line at the same station,
- a transfer activity $\left(a_{l_1}^s, d_{l_2}^s\right) \in \mathcal{A}_{\text{transfer}} \subset \mathcal{A}$ at station s that links the arrival event $a_{l_1}^s$ of line l_1 at station s to the departure event $d_{l_2}^s$ of a different line l_2 at the same station s.

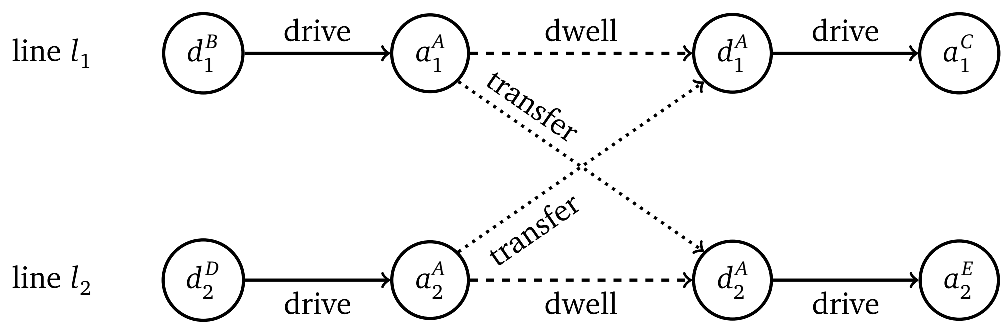

Figure 4.1: An example of an event-activity network with two lines l_1 and l_2, and five stations A, B, C, D, and E of which station A is the transfer station. A passenger can start from a certain departure event and end at a certain arrival event. The solid, dashed and dotted arcs are in the sets $\mathcal{A}_{\text{drive}}$, $\mathcal{A}_{\text{dwell}}$, and $\mathcal{A}_{\text{transfer}}$ respectively.

To illustrate the event-activity network, Figure 4.1 shows an example of this network. This example has two lines, line l_1 and l_2 and has five stations A, B, C, D, and E. Station A is the transfer station. For example, a passenger from station B to station E can travel using the path $d_1^B \to a_1^A \to d_2^A \to a_2^E$. The passenger then transfers at station A and uses the transfer activity from the arrival event a_1^A to the departure event d_2^A.

4.2 Cycle Periodicity Formulation

The Cycle Periodicity Formulation (CPF) is an alternative formulation that is transformed from the PESP formulation and is introduced by Nachtigall (1996). In this formulation, the time information is not linked to the events of the EAN but linked to the activities in the EAN. Since we are interested in whether the cycles are valid for the timetable and not the exact times of the timetable, we use this formulation in order to check whether a timetable is feasible. To ensure that the line plan and transfers $(\mathcal{L}, f, z)$, obtained by solving the line planning problem with transfer decisions, provide a feasible timetable, we need to introduce some extra definitions. Consider a directed graph G, we define a cycle in a graph as follows.

Definition 5. A *cycle* $C = (v_1, \ldots, v_k, v_1)$ is a path in a graph $G = (\mathcal{E}, \mathcal{A})$ that visits events $e_1, \ldots, e_k \in \mathcal{E}$ and returns to $e_1 \in \mathcal{E}$.

Alternatively, a cycle can be represented as a sequence of activities, that is $(e_1, e_2) \in \mathcal{A}$ to $(e_{k-1}, e_k) \in \mathcal{A}$ and back to $(e_k, e_1) \in \mathcal{A}$. A cycle in a graph does not need to follow the direction of the arcs, and therefore we can distinguish two sets of arcs in a cycle: forward arcs C^+ and backward arcs C^-.

Next, we define the concepts of potential and tensions. Consider a directed graph $G = (\mathcal{E}, \mathcal{A})$, then a *potential* is defined as a function $\pi_i : \mathcal{E} \rightarrow \mathbb{R}$. A *tension* is the set of arc values τ_a with $a \in A$ and is defined as a function $\tau : \mathcal{A} \rightarrow \mathbb{R}$ and this function relates to some potential π as follows:

$$\tau_a = \pi_j - \pi_i \qquad \forall a = (i,j) \in \mathcal{A}. \tag{4.7}$$

The periodic variant of potentials and tensions can be defined as follows. The *periodic potential with period T* is defined as a function $\pi_T : \mathcal{E} \rightarrow \mathbb{R}$, with the values of $\pi_T \in [0, T)$ for all $i \in \mathcal{E}$. The corresponding *periodic tension with period T* is a function $\tau_T : \mathcal{A} \rightarrow \mathbb{R}$, with $\tau_T \geq 0$ and periodic potential π_T with period T and $p_a \in \{0, 1\}$.

$$\tau_{a,T} = \pi_{j,T} - \pi_{i,T} + T p_a \qquad \forall a = (i,j) \in \mathcal{A}. \tag{4.8}$$

Then, consider the concept of cycle periodicity.

Definition 6. The *cycle periodicity property* for a cycle time T holds for cycle C, a set of arc values τ_a, $a \in A$, if for some *cycle periodicity integer variable q_C*,

$$\sum_{a \in C^+} \tau_{a,T} - \sum_{a \in C^-} \tau_{a,T} = T q_C, \tag{4.9}$$

where $\tau_{a,T}$, with $a \in C$, is the periodic tension for arc a with cycle period T.

Nachtigall (1996) proved the following theorem regarding the cycle periodicity property. We define the set $\mathcal{C}$ as the set of all cycles in the graph G.

Theorem 1. Given a directed graph $G = (\mathcal{E}, \mathcal{A})$ and a period T, a set of non-negative tensions τ_a, $a \in \mathcal{A}$, there is a periodic tension if and only if, there exists an integer variable q_C for each cycle in $C \in \mathcal{C}$, such that,

$$\sum_{a \in C^+} \tau_{a,T} - \sum_{a \in C^-} \tau_{a,T} = T q_C. \tag{4.10}$$

Lemma 1. The cycle periodicity integer variable q_C relates to the PESP integer variable p_a as follows:

$$q_C = \sum_{a \in C^+} p_a - \sum_{a \in C^-} p_a, \tag{4.11}$$

where $a = (i,j) \in \mathcal{A}$.

For a proof, we refer to Peeters (2003). We can now provide the CPF, which is related to the PESP.

$$
\begin{aligned}
(CPF) \qquad & \min && F(\tau_T, q) && && (4.12)\\
& \text{s.t.} && \sum_{a \in C^+} \tau_{a,T} - \sum_{a \in C^-} \tau_{a,T} = T q_C. && \forall C \in \mathcal{C}, && (4.13)\\
& && l_a \leq \tau_{a,T} \leq u_a && \forall a \in \mathcal{A}, && (4.14)\\
& && q_C \in \mathbb{Z} && \forall C \in \mathcal{C}. && (4.15)
\end{aligned}
$$

Here, constraints (4.13) ensure that the cycle periodicity property holds for every cycle in the graph. Constraints (4.14) are the time window constraints for every arc. Constraints (4.15) ensure that the cycle periodicity variable is integer.

4.3 Example

To illustrate the cycle periodicity property, we provide an example network with four lines $\mathcal{L} = \{l_1, l_2, l_3, l_4\}$ and six stations $\mathcal{S} = \{A, B, C, D, E, F\}$. The graph of Figure 4.2 shows this network. The dashed, dashed-dotted, dotted, and solid lines depict respectively l_1, l_2, l_3, and l_4. Stations A, B, C, and D are transfer stations as these stations serve more than one line. Note that, station E is not a transfer station as this station has just one edge.

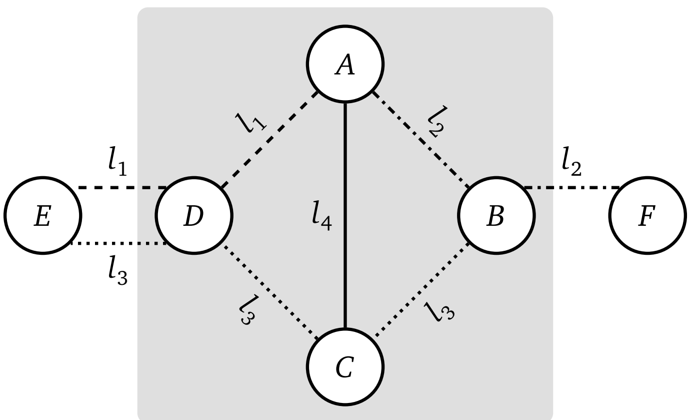

Figure 4.2: An example of a public transport network with four lines and six stations.

Let the event-activity network $\mathcal{N} = (\mathcal{E}, \mathcal{A})$ be a graph with constraint arcs. That is, each activity has a lower and an upper bound that represents the minimal and maximal duration of the activity. Recall that an event-activity network is a graph with vertices and arcs that represent events and activities respectively. We assume that we have a cycle period of

$T = 60$. In Figure 4.3, the event-activity network is shown for the gray area of the public transport network from Figure 4.2. The vertices represent the events and there are two types of events: arrival and departure events. For each line, there are forward and backward vertices, as a line usually runs in both directions. The forward and backward departure (arrival) events are denoted by $\overrightarrow{d_l^s}$ and $\overleftarrow{d_l^s}$ ($\overrightarrow{a_l^s}$ and $\overleftarrow{a_l^s}$) respectively, where l is the line and s is the station. The solid arcs are driving or dwelling arcs and the dash-dotted arcs are the transfer arcs. Note that, for station D, we do not have transfer arcs from the arrival event $\overrightarrow{a_1^D}$ to departure event $\overleftarrow{d_3^D}$ and arrival event $\overrightarrow{a_3^D}$ to departure event $\overleftarrow{d_1^D}$, as this transfer would mean that the passenger taking this arc would return to station E. Line 1 and line 3 share the same stations at this edge in the PTN, see Figure 4.2.

Table 4.1: List of cycles in the example.

Cycle	Path	Length	Direction
C_1	$\overrightarrow{d_2^A} \to \overrightarrow{a_2^B} \to \overrightarrow{d_3^B} \to \overrightarrow{a_3^C} \to \overleftarrow{d_3^C} \to \overleftarrow{a_3^D} \to \overrightarrow{d_1^D} \to \overrightarrow{a_1^A} \to \overrightarrow{d_2^A}$	60	clockwise
C_2	$\overleftarrow{d_1^A} \to \overleftarrow{a_1^D} \to \overrightarrow{d_3^D} \to \overrightarrow{a_3^C} \to \overrightarrow{d_3^C} \to \overrightarrow{a_3^B} \to \overleftarrow{d_2^B} \to \overleftarrow{a_2^A} \to \overleftarrow{d_1^A}$	60	counterclockwise
C_3	$\overrightarrow{d_2^A} \to \overrightarrow{a_2^B} \to \overrightarrow{d_3^B} \to \overrightarrow{a_3^C} \to \overrightarrow{d_4^C} \to \overleftarrow{a_4^A} \to \overrightarrow{d_2^A}$	$35 + x$	clockwise
C_4	$\overleftarrow{d_4^A} \to \overleftarrow{a_4^C} \to \overrightarrow{d_3^C} \to \overrightarrow{a_3^B} \to \overleftarrow{d_2^B} \to \overleftarrow{a_2^A} \to \overrightarrow{d_4^A}$	$35 + x$	counterclockwise
C_5	$\overrightarrow{d_4^A} \to \overleftarrow{a_4^C} \to \overleftarrow{d_3^C} \to \overleftarrow{a_3^D} \to \overrightarrow{d_1^D} \to \overrightarrow{a_1^A} \to \overrightarrow{d_4^A}$	$35 + x$	clockwise
C_6	$\overleftarrow{d_1^A} \to \overleftarrow{a_1^D} \to \overrightarrow{d_3^D} \to \overrightarrow{a_3^C} \to \overrightarrow{d_4^C} \to \overrightarrow{a_4^A} \to \overleftarrow{d_1^A}$	$35 + x$	counterclockwise

We now first consider only lines l_1, l_2, and l_3. We ignore line l_4 and the events corresponding to line l_4. In Table 4.1, the paths of the cycles are shown as well as the length of the cycles. In this case, we have two cycles C_1 and C_2, the other cycles contain line 4. Note that, C_1 and C_2 are in fact parallel cycles but C_1 and C_2 are in the opposite direction. C_1 is clockwise and C_2 is counterclockwise following the direction of the arcs. As both lines have a length of 60 and this length is a multiple of the cycle period, there exists a feasible solution to the CPF.

We now consider the extra line l_4 and the corresponding events. Let the duration of the activities with an x be 10. Now, we have six cycles. The original cycles, C_1 and C_2, and the additional cycles C_3, C_4, C_5, and C_6. In Table 4.1, the length of the additional cycles are shown. As one can observe, if $x = 10$, then the length of cycles C_3, C_4, C_5, and C_6 is now 45, which is not a multiple of 60. Therefore, if we have a line plan with these four lines and the current short transfers, then there is no feasible timetable for this line plan.

Let $x = 25$, then the length of the additional cycles is 60, which is a multiple of the cycle period. The length of the third cycle is now 60 and this is a multiple of 60. Therefore, if $x = 25$, then there exists a feasible timetable for this line plan with the short transfers.

To summarize this, in order to find a feasible timetable, all cycles that are in the event-activity network obtained by the line concept should be a multiple of the cycle period T. If this is not the case, constraints (4.13) can not be met. That is, if the left hand side in Equation (4.13) is not a multiple of T, the cycle periodicity integer variable q_C for cycle C is not integer and that violates constraint (4.15).

4.4 Additional constraints to the model formulation

In order to obtain a line plan and transfer decisions, additional constraints have to be added to the model formulation provided in Chapter 3. The cycle periodicity property has to be met, that is, in the graph with short transfers we can not have any cycles that violate this property. To translate the model formulation from line planning in Chapter 3 to the EAN in the timetabling, we consider the following arcs:

- travel arcs between every adjacent pair of the station-line vertices for every line and in both directions with a riding time,
- an arc from every station-line vertex to every station-line vertex for short transfer with a short travel time.

In timetabling, we have three types of arcs, travel, transfer, and dwell arcs. The dwell arcs are new in the EAN compared to the graph used in line planning and in order to correspond with the line planning model, we do not assign time costs t_a to the dwell arcs.

Let A^t be the set of short transfer arcs. Let $\mathcal{C}$ be the set of all possible cycles in the graph. A subset of $\mathcal{C}_{\text{invalid}} \subseteq \mathcal{C}$ violates the cycle periodicity property. That is the case when the sum of the activities in the cycle is not a multiple of the cycle period T. Let $\delta(C)$ be the set of arcs that are in the cycle C, then $\ell(C)$ is the length of the cycle and this can be computed as follows:

$$\ell(C) = \sum_{a \in \delta(C)} t_a, \tag{4.16}$$

where $\delta(C) = \{(i,j) \in C : i \in \mathcal{E}, j \in \mathcal{E}\}$. A cycle is invalid if $\ell(C)$ is not a multiple of the cycle period T. As we do not provide a full timetable for this problem, we can allow for a

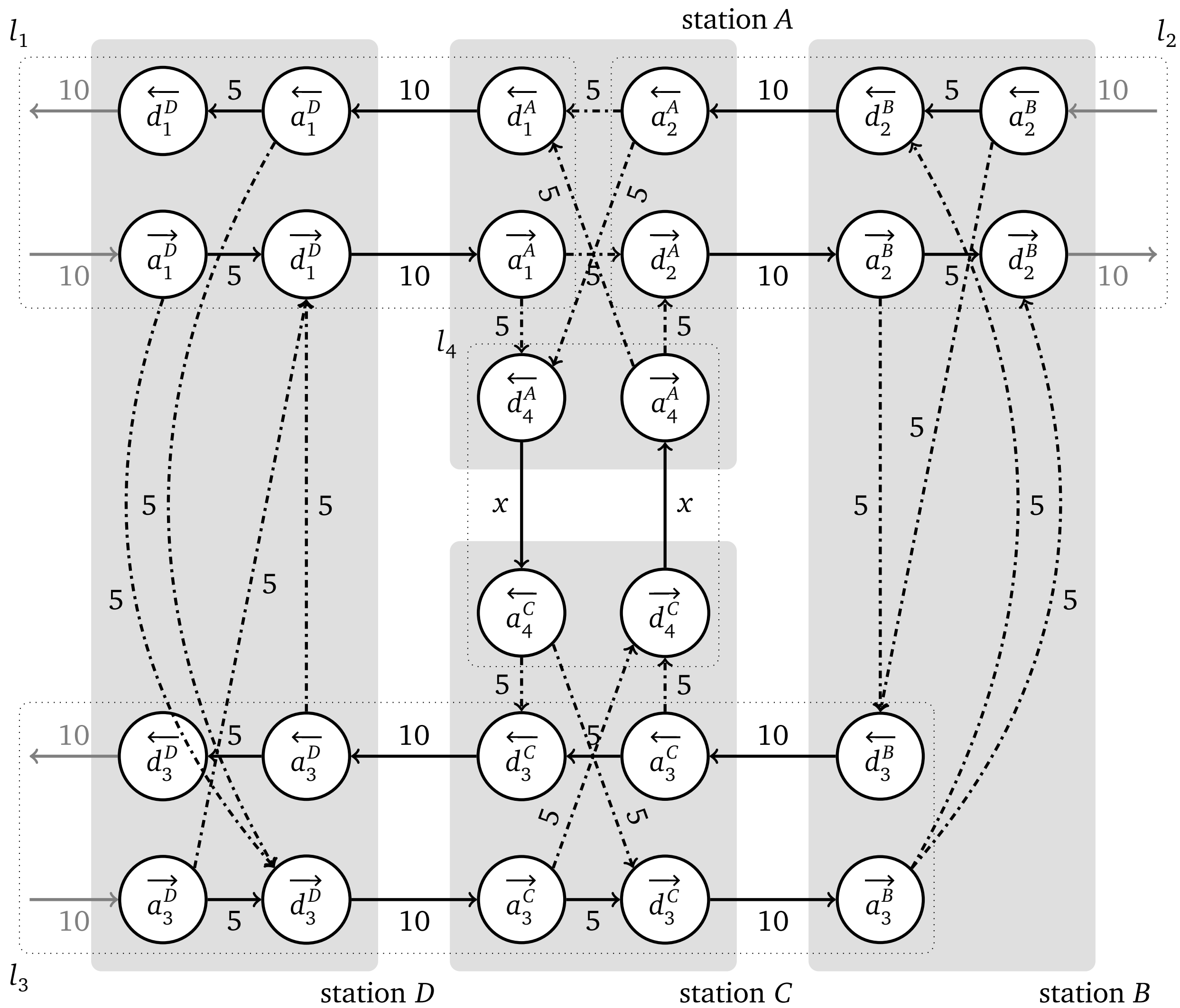

Figure 4.3: An example of an event-activity network with cycles from the public transport network in Figure 4.2. The arrival and departure events are represented by the vertices a_l^s and d_l^s, where l is the line number and s is the station name. The drive and dwell arcs are represented by solid arcs. The transfer arcs are represented by dashed-dotted arcs. The duration of the activity is shown for each arc. The transfer and dwell activity has a duration of 5, and the driving activity has a duration of 10. The duration of the driving activities of line l_4 is x.

small deviation from the cycle period T. Let Δ be that deviation, then a cycle is invalid if $\ell(C) \mod T \in [\Delta, T-\Delta]$. Thus the set of invalid cycles is:

$$\mathcal{C}_{\text{invalid}} = \left\{ C \in \mathcal{C} : \ell(C) \mod T \in [\Delta, T-\Delta] \right\}. \tag{4.17}$$

4.4.1 Transfer decisions

Now we can introduce a new set of variables and constraints in order to incorporate the transfer decisions in the line planning model formulation. Consider the set of short transfer arcs $A^t \subset A$. A transfer arc is an arc from a station-line vertex to a station-line vertex and where the station is the same but the line is different. For each transfer possibility at every transfer station, there is a transfer decision variable z_a for all arcs $a = (s^i_{l_1}, s^i_{l_2}) \in A^t$. This decision variable shows whether a short transfer from line l_1 to l_2 is offered at station i or not. That is,

$$z_a = \begin{cases} 1 & \text{if a short transfer arc } a = (s^i_{l_1}, s^i_{l_2}) \text{ is offered from line } l_1 \text{ to } l_2 \text{ at station } i, \\ 0 & \text{otherwise.} \end{cases} \tag{4.18}$$

Then, we have the following constraints for transfer decisions:

$$\sum_{o \in \mathcal{S}} y^a_o \leq M z_a, \qquad \forall a \in A^t, \tag{4.19}$$

$$z_a \in \{0,1\}, \qquad \forall a \in A^t. \tag{4.20}$$

Constraints (4.19) ensure that if a short transfer is offered at station i from l_1 to l_2, then all flow decision variables $y_o^{\left(s^i_{l_1}, s^i_{l_2}\right)}$ for all origin stations o can be greater than 0. M is a sufficiently large number such that all flows can be met if a short transfer is allowed. M is set to the total number of passengers of the instance in order to obtain a tighter bound as the flow can not exceed the total number of passengers of the instance. Constraints (4.20) ensure that the binary variable takes 0 or 1.

4.4.2 Cycles in the network

Let $\delta(C)$ be the set of arcs of cycle C and let A^t be the set of transfer arcs. If there is an invalid cycle, we want to exclude that cycle in the line planning model formulation. Therefore, we do not want to have all transfers that can be offered in the invalid cycle.

Let $\delta^t(C) = \delta(C) \cap A^t$ be the set of transfer arcs of cycle $C \in \mathcal{C}_{\text{invalid}}$, then to exclude all invalid cycles we impose the following constraints,

$$\sum_{a \in \delta^t(C)} z_a \leq |\delta^t(C)| - 1, \qquad \forall C \in \mathcal{C}_{\text{invalid}}. \tag{4.21}$$

Constraints (4.21) ensure that from the set of excluded transfer combinations not all decision variables from the set can be covered, that is, at least one of the decision variables must be zero.

As the number of invalid cycles $\mathcal{C}_{\text{invalid}}$ can be exponentially high, there are also exponentially many constraints (4.21). Enumerating this set of cycles is very hard and therefore we need to find other ways to obtain the optimal solution for the line planning with transfer decisions.

Figure 4.4 shows the graph of the example from Section 4.3 that is similar to Figure 4.3. Recall that we have six cycles as shown in Table 4.1. For each of the transfer activities, the associated transfer decision variable is shown instead of the duration of the transfer (which is 5 for all transfers) in Figure 4.4. Note that, we have set the duration of line 4 between station A and C to 10. According to the CPF, the cycles that include line 4 have to be eliminated. Therefore, the set of invalid cycles is $\mathcal{C}_{\text{invalid}} = \{C_3, C_4, C_5, C_6\}$. We do that using the additional decision variables. For each of the transfer activities, the associated set of decision variables of the transfer arcs is shown. These are shown in Table 4.2.

Table 4.2: List of cycles and the corresponding set of decision variables.

Cycle	Path	$\delta^t(C)$
C_1	$\overrightarrow{d_2^A} \to \overrightarrow{a_2^B} \to \overrightarrow{d_3^B} \to \overrightarrow{a_3^C} \to \overleftarrow{d_3^C} \to \overleftarrow{a_3^D} \to \overrightarrow{d_1^D} \to \overrightarrow{a_1^A} \to \overrightarrow{d_2^A}$	$\{z_{(2,3)}, z_{(3,1)}, z_{(1,2)}\}$
C_2	$\overleftarrow{d_1^A} \to \overleftarrow{a_1^D} \to \overrightarrow{d_3^D} \to \overrightarrow{a_3^C} \to \overrightarrow{d_3^C} \to \overrightarrow{a_3^B} \to \overleftarrow{d_2^B} \to \overleftarrow{a_2^A} \to \overleftarrow{d_1^A}$	$\{z_{(1,3)}, z_{(3,2)}, z_{(2,1)}\}$
C_3	$\overrightarrow{d_2^A} \to \overrightarrow{a_2^B} \to \overrightarrow{d_3^B} \to \overrightarrow{a_3^C} \to \overrightarrow{d_4^C} \to \overleftarrow{a_4^A} \to \overrightarrow{d_2^A}$	$\{z_{(2,3)}, z_{(3,4)}, z_{(4,2)}\}$
C_4	$\overleftarrow{d_4^A} \to \overleftarrow{a_4^C} \to \overrightarrow{d_3^C} \to \overrightarrow{a_3^B} \to \overleftarrow{d_2^B} \to \overleftarrow{a_2^A} \to \overrightarrow{d_4^A}$	$\{z_{(4,3)}, z_{(3,2)}, z_{(2,4)}\}$
C_5	$\overrightarrow{d_4^A} \to \overleftarrow{a_4^C} \to \overleftarrow{d_3^C} \to \overleftarrow{a_3^D} \to \overrightarrow{d_1^D} \to \overrightarrow{a_1^A} \to \overrightarrow{d_4^A}$	$\{z_{(4,3)}, z_{(3,1)}, z_{(1,4)}\}$
C_6	$\overleftarrow{d_1^A} \to \overleftarrow{a_1^D} \to \overrightarrow{d_3^D} \to \overrightarrow{a_3^C} \to \overrightarrow{d_4^C} \to \overrightarrow{a_4^A} \to \overleftarrow{d_1^A}$	$\{z_{(1,3)}, z_{(3,4)}, z_{(4,1)}\}$

For each of the cycles, we have a constraint (4.21). That means that for each of the cycles, at least one of the decision variables has to be 0, and thus that short transfer connection is not allowed. So by setting $z_{(4,2)} = z_{(2,4)} = z_{(4,1)} = z_{(1,4)} = 0$, the cycles in $\mathcal{C}_{\text{invalid}}$ have been eliminated and constraints (4.21) are satisfied. Therefore, a short transfer possibility is not provided to and from line 4. As a result of this, the flow arcs corresponding to these transfer decision variables are also set to zero by the constraints (4.19) that link the flow variables with the transfer variables.

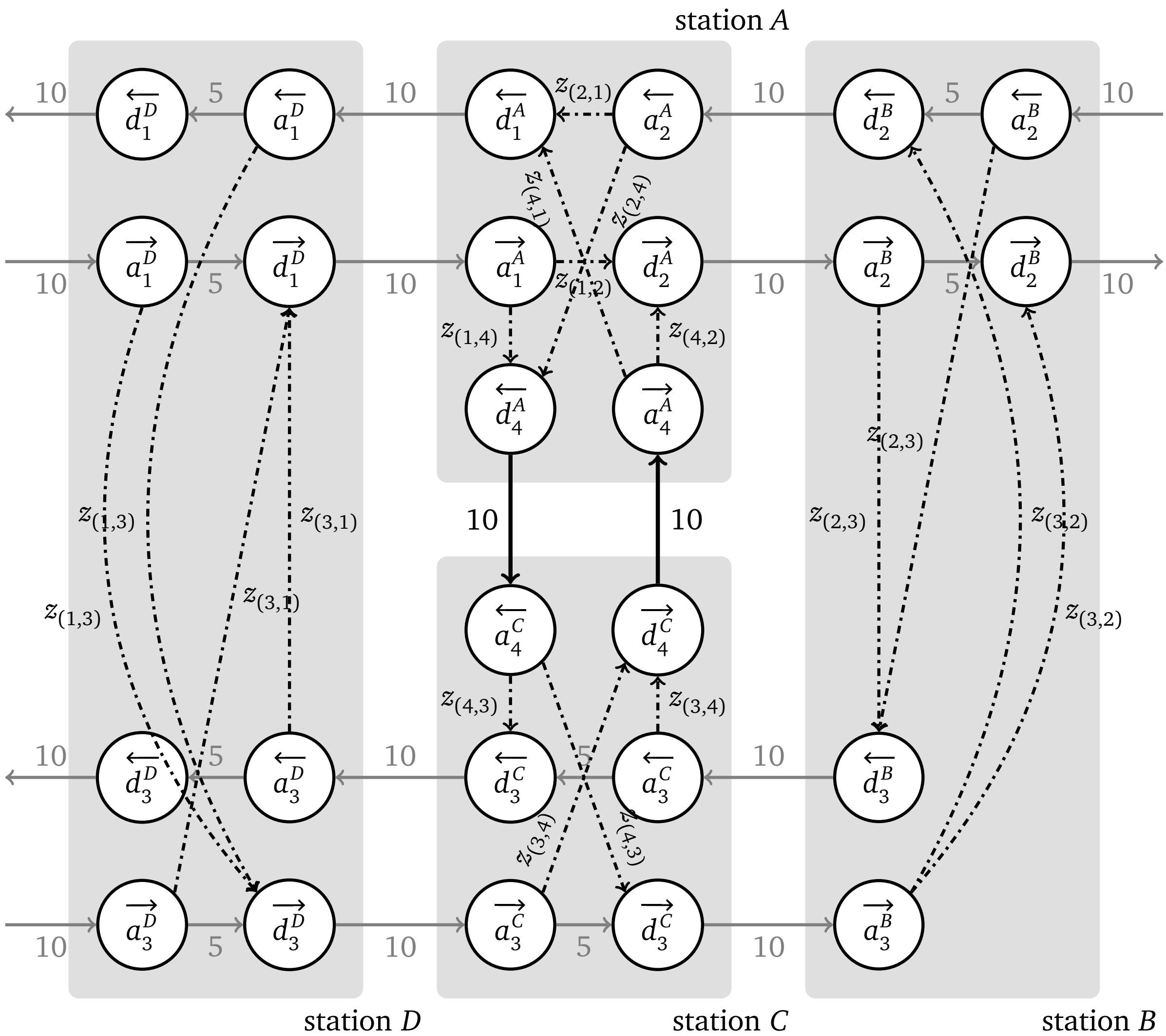

Figure 4.4: An event-activity network with six cycles. For every transfer activity, we have a decision variable that corresponds with the transfer arc in the line planning model formulation. These decision variables are denoted by $z_{(i,j)}$, where i and j are line numbers with $i \neq j$. The duration of all transfer arcs is 5 and is not shown to keep the figure readable.

Chapter 5

Constraint Generation

In Chapter 3 and 4, the focus was on the problem description. This chapter focuses on the solution method we use to find a satisfactory solution for line planning with transfer decisions.

5.1 Relaxing the problem

The model formulation of line planning and the additional constraints given in Chapter 4 are used to find a satisfactory solution. However, there are some limitations to this model. The model formulation is a combination of a line selection problem and a multi-commodity flow problem. The additional constraints are comparable to the subtour elimination constraints of the traveling salesman problem. This problem is, therefore, $\mathcal{NP}$-complete, as the multi-commodity flow problem is $\mathcal{NP}$-complete. $\mathcal{NP}$-complete means that the solving time increases quickly if the size of the problem grows. However, a solution to the problem can be verified quickly.

As our problem has an exponential number of constraints, we apply constraint generation to the MILP of the line planning formulation provided in Chapter 3 with the additional constraints in Chapter 4. We relax this problem by removing constraints (4.21). Then, we solve the relaxation of the problem using branch-and-bound and when the optimal solution is found for the relaxation, then we check whether this problem is infeasible for the original problem. In this case, this means there exists a cycle in the solution that violates the cycle periodicity property. In order to resolve this, we use a cycle detection algorithm to find invalid cycles. These cycles are then added as constraints to the relaxed MILP. We continue until no invalid cycles are found. A flowchart is shown for this approach in Figure 5.1.

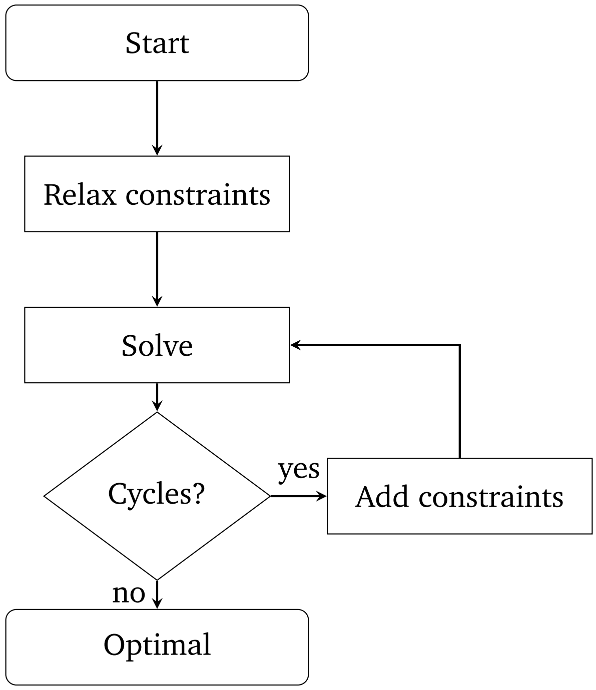

Figure 5.1: Flowchart of the algorithm to solve the line planning problem with transfer decisions.

5.2 Constraint Generation Algorithm

To obtain a solution for the line planning problem with transfer decisions, we implement a constraint generation algorithm (CGA). We solve the mathematical formulation for the line planning problem with transfer decisions (see Chapter 3). The solution obtained is a line plan with transfer decisions, denoted by $(\mathcal{L}, f, z)$, where $\mathcal{L}$ is the set of lines, f the vector of selected frequencies of the selected lines, and vector z the transfer decisions. From this solution, an event-activity network can be constructed. We refer to Figure 4.1 for an example of this network. Using this EAN, we try to find cycles that do not satisfy the cycle periodicity property with the procedure CYCLECHECK. If we find a cycle that does not satisfy the cycle periodicity property, this procedure returns a combination of transfer decisions that is not possible concurrently. This transfer combination is added to the family set of transfer combinations, which is also the input of the next iteration of optimizing the line planning problem with transfer decisions. In Algorithm 1, the pseudo-code of this algorithm is shown.

Algorithm 1 Constraint Generation Algorithm

Input: lines and frequencies

Output: optimal set of lines, frequencies and transfer decisions

procedure CGA

    $\mathcal{C}_{\text{invalid}} \leftarrow \emptyset$ ▷ set of excluded cycles

    $C_{\text{invalid}} \leftarrow \emptyset$ ▷ excluded cycle

    $(\mathcal{L}^*, f^*, z^*) \leftarrow \text{LinePlanningProblem}(\mathcal{L}, f, \mathcal{C}_{\text{invalid}})$ ▷ optimal solution of the relaxed problem

    $C_{\text{invalid}} \leftarrow \textsc{CycleCheck}((\mathcal{L}^*, f^*, z^*))$ ▷ find cycles

    while $C_{\text{invalid}} \neq \emptyset$ **do**

        $\mathcal{C}_{\text{invalid}} \leftarrow \mathcal{C}_{\text{invalid}} \cup \{C_{\text{invalid}}\}$

        $C_{\text{invalid}} \leftarrow \emptyset$

        $(\mathcal{L}^*, f^*, z^*) \leftarrow \text{LinePlanningProblem}(\mathcal{L}, f, \mathcal{C}_{\text{invalid}})$

        $C_{\text{invalid}} \leftarrow \textsc{CycleCheck}((\mathcal{L}^*, f^*, z^*))$

5.2.1 Cycle detection

The aim of CycleCheck is to detect cycles in the EAN that violate the cycle periodicity property. That is, CycleCheck tries to obtain the cycle with the largest deviation from the cycle period. The pseudo-code of CycleCheck is shown in Algorithm 2. First, we generate the EAN from the lines, frequencies and offered short transfers. We initialize the set of cycle candidates C that might be removed and set it to the empty set (line 5). In order to detect cycles, we compute the shortest path from each departure event $d_l^s \in \mathcal{E}_{\text{dep}}$ at every transfer station $s \in \mathcal{S}_{\text{transfer}}$ in the EAN $\mathcal{N}$. Then, for each of the transfer stations, a distance vector D is known (line 7). Now, we only have the shortest path and in order to obtain the shortest cycle, we need to return to the original transfer station. Therefore, we need to find the arriving vertex a_{l*}^s from which the departure event $d_l^s \in \mathcal{E}_{\text{dep}}$ has an incoming transfer or driving activity. Recall that $\ell(C)$ is the cycle length of cycle C, that is the distance calculated by Dijkstra's shortest path algorithm and the duration of the transfer activity $(a_{l*}^s, d_l^s) \in \mathcal{A}_{\text{transfer}}$ together, thus

$$\ell(C) = D[a_{l*}^s] + t_{(a_{l*}^s, d_l^s)}. \tag{5.1}$$

Let $r(C) = \ell(C)$ modulo T be the remainder of this cycle length, where T is the cycle period. The cycle has the largest deviation if the remainder $r(C) = \dfrac{T}{2}$. Let Δ be the maximum deviation for which we allow that the cycle is still valid. Thus, the candidate cycle that has to be excluded is the cycle with the remainder that deviates the most from the cycle period T. Let dev(C) be the deviation of cycle C from the cycle period T, then dev(C) is defined as follows:

$$\text{dev}(C) = \frac{T}{2} - \left| \frac{T}{2} - r(C) \right|. \tag{5.2}$$

The range of dev(C) is $\left[0, \frac{T}{2}\right]$. The calculation of $r(C)$ and dev(C) are shown in line 10 and 11.

As we do not seek to obtain a timetable, we do not reject all cycles that deviate from the cycle property. We set a parameter Δ and only the cycles that have a deviation larger than Δ will be candidates for exclusion (line 12). To obtain a cycle, we connect the departure event d_l^s with the arrival event a_{l*}^s. There exists an arc from a_{l*}^s to d_l^s as s is a transfer station. In line 14, we add C to the set of cycles $\mathcal{C}$.

We add all invalid cycles found with this procedure and exclude them at the next iteration of the constraint generation algorithm. If no invalid cycles are found, then we have found the optimal heuristic solution.

The cycle detection algorithm only follows arcs in the forward direction. However, there are also cycles that contain arcs in the backward direction. Therefore, this approach might not find all cycles that violate the cycle periodicity property. As a result of this, this approach is a heuristic. This means we might not obtain a feasible solution for the formulation provided in Chapter 3 and the additional constraints provided in Section 4.4 as we do not detect cycles that also contain backward arcs. This means that the constraint generation algorithm approximates the optimal solution and therefore the approximate solution might not provide a feasible timetable.

Dijkstra's shortest path algorithm

To find the shortest cycle, we use Dijkstra's shortest path algorithm (Dijkstra, 1959) to find the shortest paths in a graph from the starting location, also known as the source, to all other locations. Algorithm 3 shows the pseudo-code of Dijkstra's shortest path algorithm.

In this case, the weighted graph is the EAN, with the events as vertices and activities as arcs. The activities have weights and these represent the duration of an activity. The departure event e_0 is the source event. Then for every event $e \in \mathcal{E}$ in the network $\mathcal{N}$, we

Algorithm 2 Cycle detection

procedure CYCLECHECK($(\mathcal{L}^*, f^*, z^*)$)
  Input: PTN, satisfactory solution $(\mathcal{L}^*, f^*, z^*)$
  Output: set of transfer combinations to be excluded
  EAN $\leftarrow \mathcal{N} = (\mathcal{E}, \mathcal{A})$
  $\mathcal{C} \leftarrow \varnothing$ ▷ set of invalid cycles
  for $d_l^s \in \mathcal{E}_{\text{dep}}$ with $s \in \mathcal{S}_{\text{transfer}}$ **do**
    $D \leftarrow$ DIJKSTRAALGORITHM($\mathcal{N}, d_l^s$)
    for $a_{l*}^s \in \mathcal{E}_{\text{arr}}$ **do**
      $\ell(C) \leftarrow D[a_{l*}^s] + t_{(a_{l*}^s, d_l^s)}$
      $r(C) \leftarrow \ell(C)$ modulo T
      $\text{dev}(C) \leftarrow \dfrac{T}{2} - \left|\dfrac{T}{2} - r(C)\right|$
      if $\text{dev}(C) > \Delta$ **then**
        $C \leftarrow$ DIJKSTRASHORTESTPATH($\mathcal{N}, d_l^s, a_{l*}^s$) ▷ backtracking
        $\mathcal{C} \leftarrow \mathcal{C} \cup \{C\}$
  return $\mathcal{C}$

initialize the current duration $d(e) = \infty$ from event e_0 to event e and set current predecessor label $p(e) = \varnothing$ for event e. We also add every event in $\mathcal{E}$ to the set of unvisited events $\mathcal{U}$. We define the duration from our departure event $d(e_0) = 0$.

While the set of unvisited events $\mathcal{U}$ is not empty, we find the event $u \in \mathcal{U}$ with the shortest current duration $d(u)$ to the source. Event u is removed from the set $\mathcal{U}$. Then, for every adjacent event of event u, $v \in \mathcal{U}$, we calculate the duration $d^* = d(u) + w(u, v)$, where $w(u, v)$ is the weight of the arc (activity) between u and v. If the new found duration d^* is shorter than the current duration from the source event to event v, $d(v)$, then we set $d(v)$ to the new found shortest duration d^* and the predecessor label of event v, $p(v)$, is set to event u.

In order to obtain the shortest path, we need a sink event e_k. We add that sink event to the path P. From that sink event, we select the predecessor of e_k, $p(e_k)$. We continue as long as the predecessor is defined, otherwise, we have found the source event e_0. The pseudocode of this procedure can be found in Algorithm 3 under the procedure DIJKSTRASHORTESTPATH.

Algorithm 3 Dijkstra's shortest path algorithm

procedure DIJKSTRAALGORITHM(EAN, source)
    Input: $\mathcal{N} = (\mathcal{E}, \mathcal{A})$, source $e_0 \in \mathcal{E}$
    Output: list of minimal distances from e_0 to every event in $\mathcal{E}$
    $\mathcal{U} \leftarrow \varnothing$ ▷ set of unvisited vertices
    for $e \in \mathcal{E}$ **do**
        $d(e) \leftarrow \infty$ ▷ set current distance from source to infinity
        $p(e) \leftarrow \varnothing$ ▷ set current predecessor label to empty
        $\mathcal{U} \leftarrow \mathcal{U} \cup \{e\}$
    $d(e_0) \leftarrow 0$ ▷ set distance from source to 0
    while $\mathcal{U} \neq \varnothing$ **do**
        $u \leftarrow \arg\min_{e \in \mathcal{U}} d(e)$
        $\mathcal{U} \leftarrow \mathcal{U} \setminus \{u\}$
        for adjacent $v \in \mathcal{U}$ of vertex u **do**
            $d^* \leftarrow d(u) + w(u, v)$
            if $d^* < d(v)$ **then**
                $d(v) \leftarrow d^*$
                $p(v) \leftarrow \{u\}$
procedure DIJKSTRASHORTESTPATH(EAN, source, sink)
    Input: $\mathcal{N} = (\mathcal{E}, \mathcal{A})$, source $e_0 \in \mathcal{E}$, sink $e_k \in \mathcal{E}$
    Output: shortest path
    $P \leftarrow \varnothing$ ▷ add sink to the path
    $q \leftarrow p(e_k)$ ▷ obtain predecessor sink event
    while $q \neq \varnothing$ **do**
        insert q to path P
        $q \leftarrow p(q)$

Chapter 6

Computational Results

This chapter focuses on the computational results of the method to solve the line planning problem with transfer decisions. First, we describe the test instances and then we provide the computational results from these test instances.

6.1 Test instances

LinTim is the data set created by A. Schiewe et al. (2020). The authors provide artificial data and data based on the real world. The artificial data instances are called `toy` and `grid`. The real-world data instance is from Athens' metro network (`athens`). In Table 6.1, a selection of the instances used is shown along with the instance's characteristics, such as the number of stops, edges, and lines. Also, the number of OD pairs and the number of passengers in the system are shown.

Table 6.1: An overview of the instances.

instance	type	size				
		stops	edges	lines	OD pairs	passengers
`toy`	artificial	8	8	8	22	2622
`grid`	artificial	25	40	n/a	567	2546
`athens`	real world metro	51	52	59	2385	63323

`toy` instance

The `toy` instance is a small artificial instance with 8 stops, edges, and lines. This instance is used for testing. This instance was not useful for the analysis as the instance is too small to obtain insightful results.

`grid` instance

The `grid` instance is an artificial 5×5 grid. This instance has 25 stops and 40 edges. No lines are provided and there are 567 OD pairs. Figure 6.1 shows the grid instance. For every travel arc in the grid, that is an arc between two stations, we have a fixed duration of 8. This instance does not provide a line pool and we, therefore, generated line pools ourselves. We generated three sets of lines, named 5H, 5V, and 4D. We have 5 horizontal lines which are denoted by 5H. Next to these lines, we have 5 vertical lines denoted by 5V. Next to horizontal and vertical lines, we added diagonal zig-zag lines from the four corners of the grid, denoted by 4V. In this case, diagonal means that we start from the corner and iteratively go one station to the right, and then one down. Another diagonal zig-zag line is obtained by iteratively going one station down first and then going to the right. Table 6.2 shows the different line pools for this instance. Each of the lines has a frequency of 1 to 4 if selected. The largest line pool is 5H5V4D, which has 14 lines. Line pools 5H5V, 5H4D, and 5V4D are subsets of the line pool 5H5V4D. In Appendix B, the OD matrix is provided in Table B.1 and the routes of each line are provided in Table B.2.

Table 6.2: This table shows the line pools used as input for the 5×5 grid.

name	$\lvert\mathcal{L}\rvert$	description
5H5V	10	5 horizontal and 5 vertical lines
5H4D	9	5 horizontal and 4 diagonal zig zag lines
5V4D	9	5 vertical and 4 diagonal zig zag lines
5H5V4D	14	5 horizontal, 5 vertical and 4 diagonal zig zag lines

In order to test the algorithm on more instances, we adapt this instance to a 4×4 grid. We removed 9 stations from the instance, these stations are the 5 stations at the bottom and the 4 remaining stations at the right. The edges to and between these stations are also

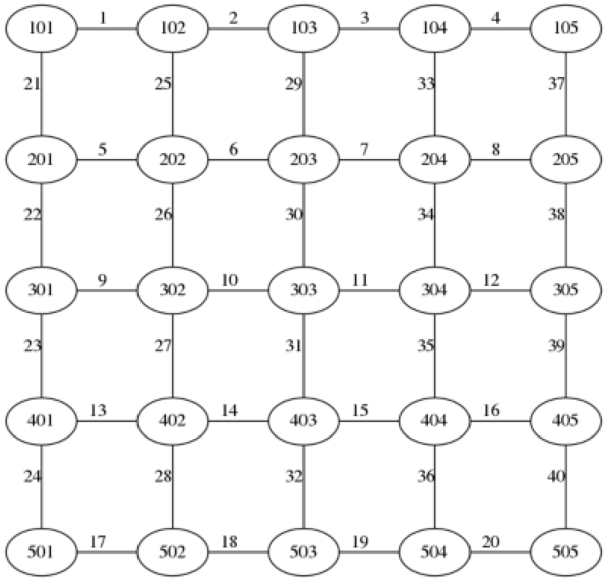

Figure 6.1: The PTN of the `grid` instance. This graph is taken from A. Schiewe et al. (2020).

removed and consequently the OD pairs from, to, and between these stations are removed. For this instance, we also have three sets of lines: 4H, 4V, and 4D. 4H, 4V, and 4D are respectively the set with 4 horizontal, vertical and diagonal lines. In Table 6.3 the line pools are shown for the 4×4 grid. In Table B.3 of the appendix, the exact routes of the lines are shown.

Table 6.3: This table shows the line pools used as input for the 4×4 grid.

name	$\lvert\mathcal{L}\rvert$	description
4H4V	8	4 horizontal and 4 vertical lines
4H4D	8	4 horizontal and 4 diagonal zig zag lines
4V4D	8	4 vertical and 4 diagonal zig zag lines
4H4V4D	12	4 horizontal, 4 vertical and 4 diagonal zig zag lines

`athens` instance

The instance of Athens metro consists of 51 stops, 52 edges, and 59 lines with 2385 OD pairs. As this instance results in a long solving time for MIP, we reduced the number of stops, edges, lines, and OD pairs by cutting the tails of every line. After reducing, there are 14 stops, 15 edges, 14 lines, and 182 OD pairs remaining.

Athens' metro network consists of three main lines and those lines come together in the city center. Figure 6.2 shows the structure of the reduced PTN. In this figure, the stations omitted are the ones from the dotted arc at both ends of the lines. We remove the stations after the first non-transfer station at both ends of each line. The OD pairs within the omitted section of a line are removed. OD pairs from an omitted station to a station in the reduced PTN are considered as if the passenger originates from the first station along the line in the reduced PTN (and vice versa). If the OD pair originates from an omitted station and ends at an omitted station, but there exists a section that is in the reduced PTN, then the OD pair is changed to an OD pair from the first station in the reduced PTN to the last station before the cut-off.

Lines within the PTN are removed if the entire line is in the omitted section. If the line originates from an omitted station and ends at a station included, then only the section of the line is considered that is within the reduced PTN.

6.2 Parameter settings

The model and the algorithm include several parameters that can be changed. The maximum line budget $c_{\max}$ can be varied in order to obtain different line plans and thus different transfer possibilities. The minimal deviation from the cycle period Δ has been set to 3. Table 6.4 shows an overview of the parameters used.

Table 6.4: This table shows the parameters used.

symbol	parameter	value
$c_{\max}$	maxmimal total line costs	experiment specific
t_{short}	short transfer time	2
Δ	minimal deviation from cycle period	3

6.3 Results for the line planning with transfer decisions

The code is written in Java and the formulations are solved by CPLEX solver version 20.1. This solver is provided by an educational license of IBM ILOG CPLEX Optimization Studio.

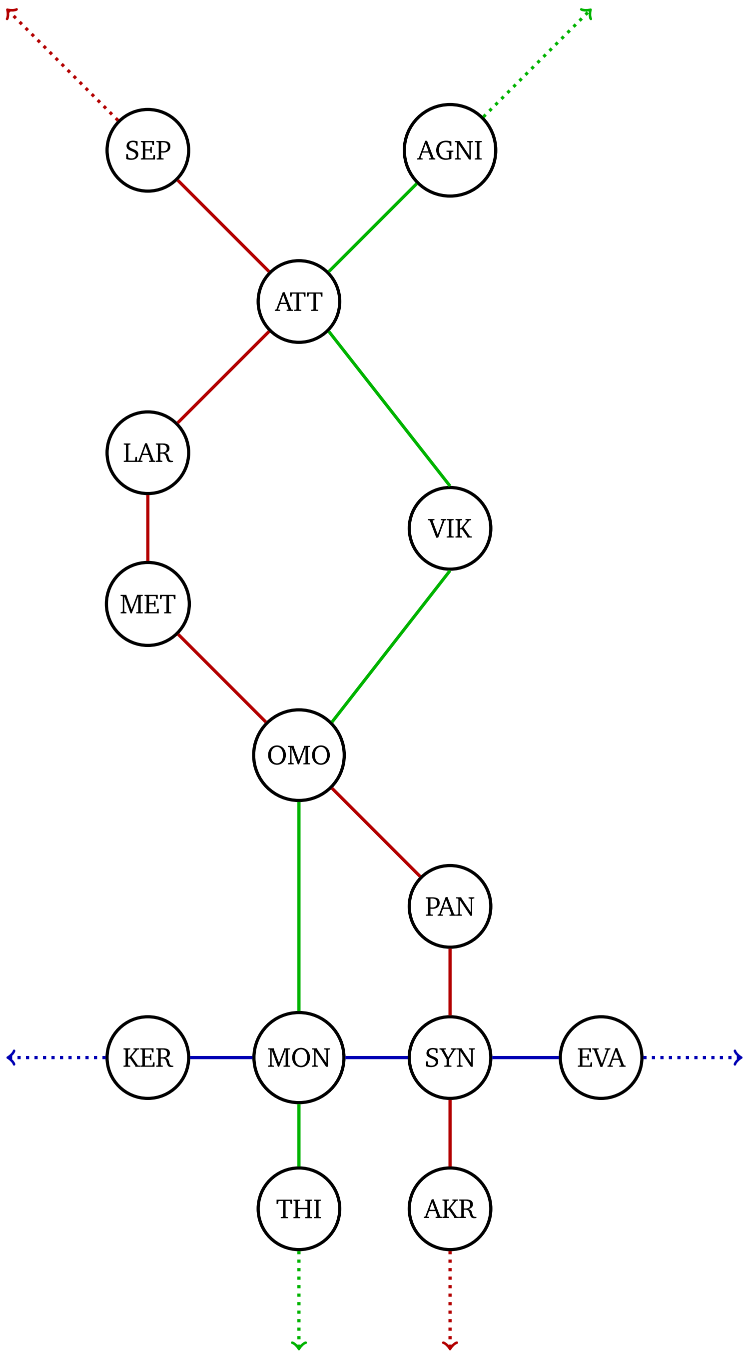

Figure 6.2: A schematic view of the Athens metro network. In the reduced network, line 1 (green) starts from AGNI and ends at THI, line 2 (red) starts from SEP and ends at AKR, and line 3 (blue) starts from KER and ends at EVA. On both sides of the lines, the stops are omitted from the first non-transfer station after the last transfer station.

The computational experiments are run on a Windows 11 laptop with an AMD Ryzen 7 4800U processor and 16 GB of RAM.

6.3.1 Comparing line planning with transfer decisions

We first compare the line planning problem with transfer decisions with the line planning problem without offering transfer decisions. In Table 6.5 the results are shown for the 4×4 grid and the `athens` instance. When short transfers are included, the objective value is lower for all maximum line cost values. However, for some instances, the solving time is longer and this is mainly due to the larger number of constraints. The line costs of the `grid` instance are 10 and the line costs of the `athens` instance are provided in the data. This comparison shows that the line planning problem with transfer decisions indeed leads to overall shorter travel times and therefore it is useful to include these transfer decisions in the line planning. In the following sections, more detailed results for each instance are presented.

Table 6.5: The objective values (minimum total travel time) and running time of line planning with transfer decisions and without transfer decisions. For the 4×4 grid (left) we show the results with budget $c_{\max} \in [70, 120]$ and for the `athens` instance (right) we used a selection of interesting budgets $c_{\max}$.

(a) 4×4 grid with line pool 4H4V4D.

$c_{\max}$	short transfers			
	included		excluded	
	obj.	time (s)	obj.	time (s)
70	28,474	17.70	31,396	0.47
80	28,228	21.60	30,796	0.35
90	28,142	17.50	30,364	0.28
100	28,094	8.78	30,109	0.34
110	28,074	6.62	29,959	0.15
120	28,058	6.13	29,839	0.05

(b) `athens` instance.

$c_{\max}$	short transfers			
	included		excluded	
	obj.	time (s)	obj.	time (s)
43	3,784,247	2.78	3,820,783	0.20
45	3,783,074	3.49	3,820,783	0.16
47	3,778,783	2.09	3,820,783	0.19
49	3,778,783	4.43	3,820,783	0.16
51	3,778,616	5.51	3,820,783	0.15
53	3,778,143	4.56	3,820,783	0.14

6.3.2 Results for the 5×5 grid

For the 5×5 grid, we run the experiments with different line pools and different budgets $c_{\max}$. The budget values are based on the number of lines in the instance. Every line has a cost of 10, so we run the experiments for every multiple of 10 until all lines can be operated. So for the line pool 5H5V, we have a line pool of 10 lines and thus we run the experiment for the budgets $c_{\max} = \{60, \ldots, 100\}$. The frequency of the lines is at most 4 an hour for every line. Table 6.6 shows the results of the instance 5H5V. The objective value is the minimum total travel time of the passengers for this instance. The time in seconds is the solving time up to satisfactory. The number of iterations is the number of iterations needed until no new invalid cycles are found. The number of lines is the number of selected lines in the final solution. The number of cycles is the total number of invalid cycles found until the last iteration. Long and short transfers are the number of passengers using a long transfer and short transfer respectively.

We can observe that the longest running time is when $c_{\max} = 80$, this is due to the tightness of the budget compared to the passenger demand. Moreover, one can observe that as the line cost budget grows, the total travel time decreases. The reason behind this decrease is that more lines can be selected and therefore either more direct connections can be offered or better transfers which decreases the total travel time of passengers in the public transport system. We can also observe that the number of invalid cycles increases when the line costs budget increases. The cause of the increased number of invalid cycles is the higher number of lines selected, which increases the number of transfers. Additionally, for the budget of 70 and 80, we also see that there are passengers that have a long transfer. It seems that the shorter travel times for a large number of passengers in the system outweigh the long transfers for a small number of passengers in the case of $c_{\max} = \{70, 80\}$.

Table 6.7 shows the results for the 5×5 grid with line pool 5H5V4D. Notable is the computation time for the run with a budget of 80. This run lasted for almost 10 hours. The cause of this is the tight budget, while there are a lot of (favorable) line combinations possible, namely $\binom{14}{8} = 3003$. For all of these instances, a long transfer is not needed and all passengers that need a transfer can have a short transfer to another line. Overall the total travel time compared to the line pool 5H5V is lower as 5H5V is a subset of 5H5V4D. Furthermore, the four diagonal lines have more stops and can cover more OD pairs directly and thus this line plan can directly bring more passengers to their destination. Note that,

Table 6.6: Results of the 5 × 5 grid instance with line pool 5H5V.

						#transfers	
$c_{\max}$	objective	time (s)	#iterations	#lines	#cycles	long	short
60	61,580	0.50	1	6	0	0	2,102
70	59,502	14.15	5	7	56	46	1,814
80	57,565	45.33	4	8	70	65	1,591
90	55,920	16.17	5	9	98	0	1,544
100	55,228	3.97	9	10	118	0	1,398

the number of (short) transfers is also lower than for the line pool 5H5V due to the more diverse set of lines.

Table 6.7: Results of the 5 × 5 grid instance with line pool 5H5V4D.

						#transfers	
$c_{\max}$	objective	time (s)	#iterations	#lines	#cycles	long	short
60	60,452	10.27	3	6	12	0	1,538
70	58,084	129.97	7	7	83	0	1,450
80	55,730	34,547.57	10	8	260	0	1,273
90	54,942	1,642.85	8	9	234	0	1,079
100	54,326	112.88	7	10	245	0	947
110	54,200	240.04	10	11	374	0	884
120	54,106	83.67	13	12	496	0	837
130	54,080	138.28	24	13	981	0	824
140	54,054	84.95	33	14	1,142	0	811

Table 6.8a and Table 6.8b show the results of the experiments with line pool 5H4D and line pool 5V4D respectively. Note that these line pools do not fit well with the demand, as there are largely more passengers that need a long transfer to reach their destination. Compared to the results of line pool 5H5V in Table 6.6, the total travel time of these sets with diagonal zig-zag lines is generally higher than for experiments with a line pool with only horizontal and vertical lines.

Table 6.8: Results of the 5×5 grid instance with line pool 5H4D (left) and 5V4D (right).

(a) Line pool 5H4D.

					transfers	
$c_{\max}$	obj.	t (s)	#it.	$\lvert\mathcal{C}\rvert$	long	short
60	66,304	4.87	7	39	0	1,832
70	59,781	36.04	10	150	109	1,521
80	59,583	80.48	12	248	109	1,422
90	59,350	40.70	15	264	104	1,335

(b) Line pool 5V4D.

					transfers	
$c_{\max}$	obj.	t (s)	#it.	$\lvert\mathcal{C}\rvert$	long	short
60	74,470	9.15	10	50	0	2,147
70	63,282	33.28	10	161	156	1,807
80	63,134	47.34	13	233	158	1,718
90	62,992	25.53	12	235	152	1,676

6.3.3 Results for the 4×4 grid

Table 6.9 shows the results for the 4×4 grid with the line pool 4H4V for the budget values $c_{\max} \in \{50, \dots, 80\}$. Solving time is reasonable and as one can see the number of iterations needed to find all cycles is also low. As in the 5×5 grid, we can also observe that the total travel time decreases when the budget increases. Furthermore, the number of short transfers decreases, which is due to the higher number of lines selected. The number of transfers shows the transfers that are advised to offer in order to reduce the total travel time for all passengers in the system.

Table 6.9: Results of the 4×4 grid instance with line pool 4H4V.

					#transfers		
$c_{\max}$	objective	time (s)	#it.	$\lvert\mathcal{C}\rvert$	arcs	long	short
50	32,624	0.32	1	0	8	0	1,156
60	31,138	1.16	3	20	12	14	1,020
70	29,916	0.92	3	30	16	0	906
80	29,398	0.70	5	52	20	0	807

In Table 6.10 the results for the line pool 4H4V4D are shown. Due to the 'difficult' diagonal lines, it takes more time to solve the problem to satisfactory, as there are more cycles in this problem. The budget with the most short transfers is where $c_{\max} = 50$. In this case, there are 28 transfers that can be recommended in the timetabling phase.

Table 6.11a and 6.11b show the results of the experiments with the four diagonal lines and either four horizontal lines or four vertical lines (line pools 4H4D and 4V4D respec-

Table 6.10: Results of the 4×4 grid instance with line pool 4H4V4D.

					#transfers		
c_{max}	objective	time (s)	#it.	$\lvert\mathcal{C}\rvert$	arcs	long	short
50	30,416	13.94	8	82	28	0	364
60	29,382	15.52	10	137	29	0	303
70	28,474	17.70	11	174	33	0	249
80	28,228	21.60	15	248	33	0	222
90	28,142	17.50	13	331	31	0	179
100	28,094	8.78	16	387	29	0	155
110	28,074	6.62	17	372	33	0	145
120	28,058	6.13	28	602	30	0	137

tively). On average the total travel time of these line pools is higher, compared to the line pool with all lines included. Compared to the line pool 4H4V, the number of passengers having short transfers is lower for the line pools with diagonal lines. However, the number of short transfer activities is higher for the line pools with diagonal lines. This is because of the longer diagonal lines that have more stops than the line pool 4H4V. As a consequence more passengers do not need to transfer.

Table 6.11: Results of the 4×4 grid instance with line pool 4H4D (left) and 4V4D (right).

(a) Line pool 4H4D.

					#transfers		
c_{max}	obj.	t (s)	#it.	$\lvert\mathcal{C}\rvert$	arcs	long	short
50	30,794	24.93	13	168	30	0	481
60	29,716	8.07	10	150	28	0	398
70	29,606	20.67	16	277	29	0	343
80	29,596	30.25	21	386	26	0	338

(b) Line pool 4V4D.

					#transfers		
c_{max}	obj.	t (s)	#it.	$\lvert\mathcal{C}\rvert$	arcs	long	short
50	30,416	2.99	6	69	27	0	364
60	30,176	7.62	9	127	26	0	340
70	30,116	2.86	7	117	23	0	310
80	30,088	1.44	7	125	27	0	296

6.3.4 Results for the `athens` instance

Figure 6.3 shows the objective value of the Athens metro for each iteration until the instance is solved to satisfactory with maximum budget costs $c_{max} = 50$. At every iteration, multiple cycles have to be excluded from the network. Excluding cycles sometimes leads to a higher objective value, as preventing some short transfers will lead to a higher travel time, either by waiting for a long transfer or traveling via another route. In Figure 6.4, one can observe that the number of passengers having a short transfer decreases as the iteration count increases. The number of passengers having a long transfer, that is, a transfer that is not officially offered, increases with that number. As the short transfer can not be offered, the objective value increases since the total travel time increases.

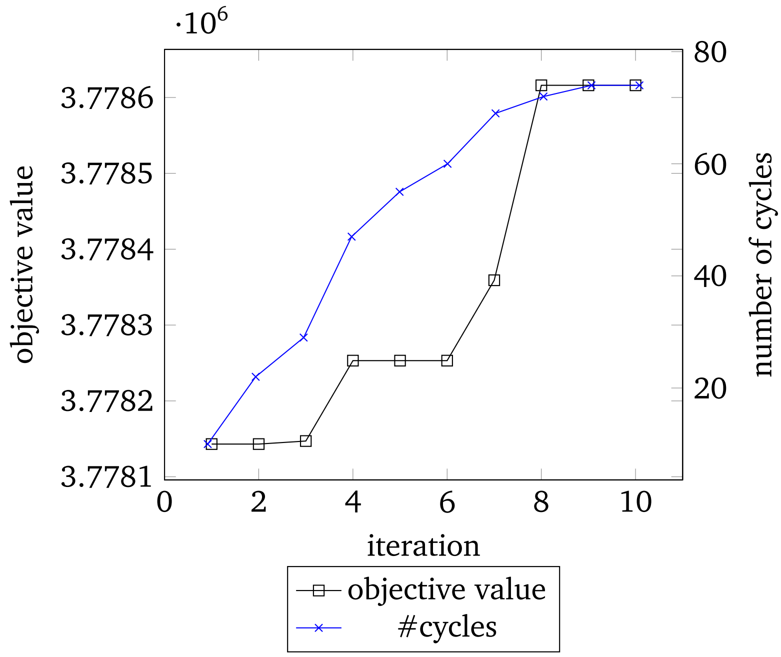

Figure 6.3: This figure shows the development of the objective value of each iteration until the most satisfactory solution is found for $c_{\max} = 50$ and the number of cycles found and eliminated until the iteration.

For this instance, the budget $c_{\max}$ needs to be at least 42 to obtain a feasible line plan with transfer decisions. In Figure 6.5, the total travel time is shown for the budgets $c_{\max} = [42, 54]$ and this figure shows that the higher the budgets are, the better the line plan can be, and therefore more short transfers can be offered. This can be observed in Figure 6.6. In Figure 6.6, the number of passengers with short and long transfers is shown for the different budgets. The largest marginal decrease of the objective value can be found at $c_{\max} = 47$. That is, if the current maximal line costs are 46, an increase of the maximum line costs of 1 yields a better overall total travel time as more short transfers can be offered.

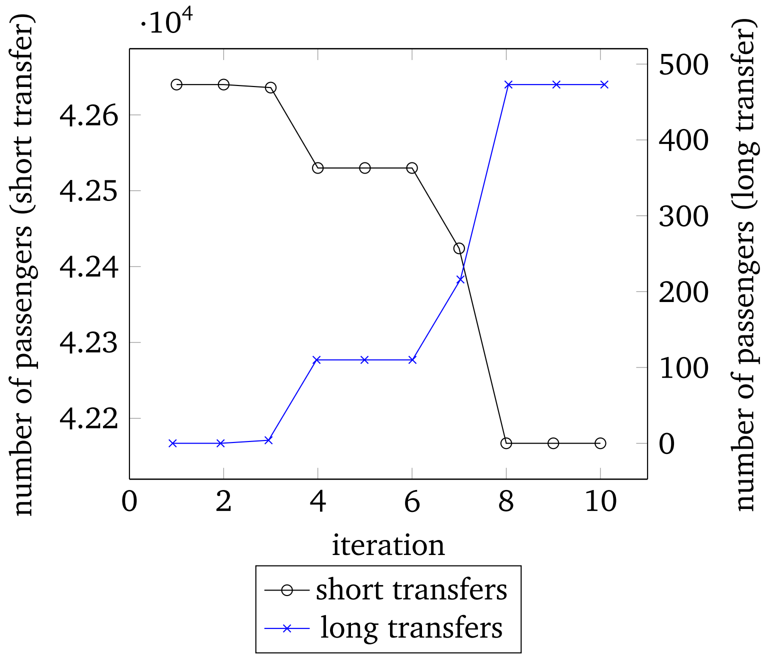

Figure 6.4: This figure shows the number of transfer passengers with a short and a long transfer for every iteration.

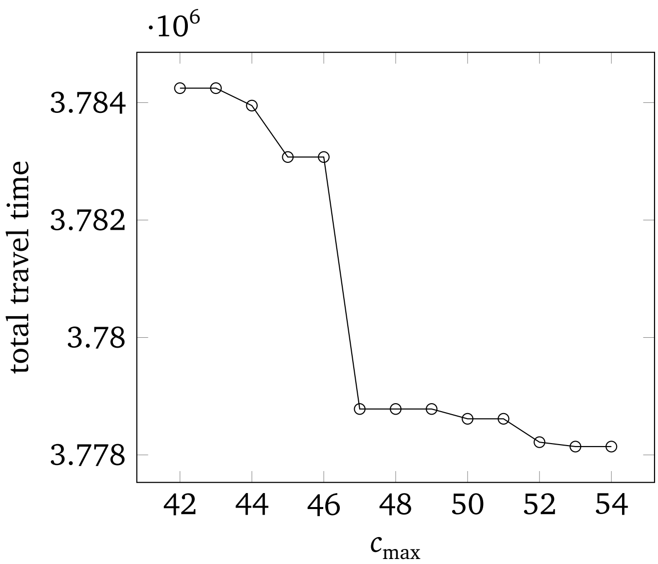

Figure 6.5: This figure shows the objective value of the most satisfactory solution with the budgets $c_{\max} \in [42, 54]$.

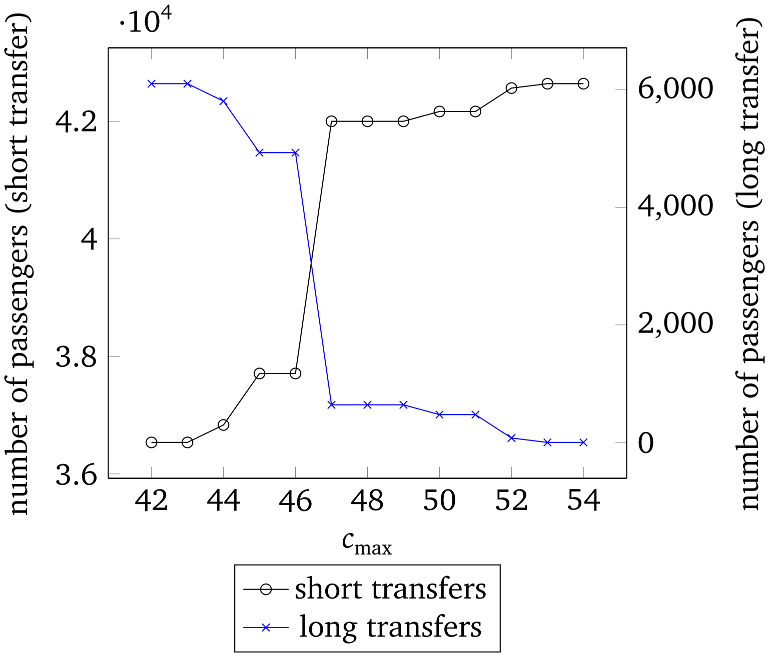

Figure 6.6: This figure shows the number of transfer passengers with a short and a long transfer with the maximum line costs for $c_{max} \in [42, 54]$.

Chapter 7

Conclusion

In this thesis, we tried to provide transfer decisions at the line planning phase of the planning process in public transport. Including transfer decisions into the line planning problem makes the problem harder to solve, as there are an exponential number of constraints that could be added to the problem to obtain a feasible solution for the line planning problem with transfer decisions. Therefore, we tried to implement a constraint generation algorithm in order to keep the number of constraints relatively small.

We tested our solution approach on an artificial data set and a real-world data set. For the real-world data set, we had to reduce the instance size to be able to obtain results in a reasonable time. For the artificial data set, we created different line pools and tried to minimize the total travel time. The results from the artificial data set show that the larger the line pool becomes, the longer the solving time is. Therefore, this is a limitation to the proposed solution method given the instance size. However, it does provide a working algorithm and we showed that it is possible to solve artificial instances as well as real-world instances and provide transfer decisions for the next planning phase in public transport planning.

One recommendation to reduce the solving time is to relax this problem further, by applying Lagrangian relaxation. One can relax the hard constraints, these are the constraints that link the different problems to each other and put these in the objective function. These linking constraints are, for example, constraints (4.19). These constraints prevent flows over the short transfer arcs if the transfer is not offered. Then the line planning problem and the transfer decisions can be solved independently from each other. These subproblems may have a shorter solution time and therefore the problem can be solved more quickly.

However, Lagrangian relaxation may need a lot of iterations which can make the solving time longer. Moreover, the solution to the Lagrangian relaxation may not be feasible for the original problem. Therefore, a heuristic can be applied to fix the violated constraints that were relaxed in the Lagrangian relaxation to obtain a feasible solution. The violation of constraints (4.19) can be overcome by moving the flows over the short transfer arcs to the corresponding long transfer arcs. Then, however, this solution can still contain invalid cycles and the Lagrangian relaxation must also be repeated with the extra cycle elimination constraints.

Another recommendation to improve the solution approach is to adapt the cycle detection algorithm in order to find all cycles in the event-activity network, which means cycles that include arcs in the backward direction. By excluding the cycles violating the cycle periodicity property that also contain backward arcs, we can ensure that the cycle periodicity property holds for all cycles remaining in the event-activity network. Therefore, we can ensure that the solution for the line planning problem with transfer decisions can provide a feasible timetable for these transfer decisions.

Conclusively, this thesis demonstrates that in the line planning phase, the transfer decisions can already be made and support the timetabling phase to create a timetable that helps to reduce the travel time in the public transport system. In the optimized line plan with transfer decisions, the passengers have shorter travel times and can have short transfers if this is possible. Shorter travel times mean that the convenience of public transport increases and thus also the attractiveness of public transport.

Bibliography

Borndörfer, R., & Neumann, M. (2010). *Models for Line Planning with Transfers* (tech. rep.). https://opus4.kobv.de/opus4-zib/frontdoor/index/index/docId/1174

Borndörfer, R., Grötschel, M., & Pfetsch, M. E. (2007). A column-generation approach to line planning in public transport. *Transportation Science*, *41*(1), 123–132. https://doi.org/10.1287/trsc.1060.0161

Borndörfer, R., & Karbstein, M. (2012). A direct connection approach to integrated line planning and passenger routing. *OpenAccess Series in Informatics*, *25*, 47–57. https://doi.org/10.4230/OASIcs.ATMOS.2012.47

Bull, S., Larsen, J., Lusby, R. M., & Rezanova, N. J. (2019). Optimising the travel time of a line plan. *4OR*, *17*(3), 225–259. https://doi.org/10.1007/s10288-018-0391-5

Cacchiani, V., & Toth, P. (2012). Nominal and robust train timetabling problems. *European Journal of Operational Research*, *219*(3), 727–737. https://doi.org/10.1016/J.EJOR.2011.11.003

Caprara, A., Kroon, L., Monaci, M., Peeters, M., & Toth, P. (2007). Chapter 3 Passenger Railway Optimization. *Handbooks in Operations Research and Management Science*, *14*(100), 129–187. https://doi.org/10.1016/S0927-0507(06)14003-7

Claessens, M. T., Van Dijk, N. M., & Zwaneveld, P. J. (1998). Cost optimal allocation of rail passenger lines. *European Journal of Operational Research*, *110*(3), 474–489. https://doi.org/10.1016/S0377-2217(97)00271-3

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik 1959 1:1*, *1*(1), 269–271. https://doi.org/10.1007/BF01386390

Goerigk, M., & Schmidt, M. (2017). Line planning with user-optimal route choice. *European Journal of Operational Research*, *259*(2), 424–436. https://doi.org/10.1016/j.ejor.2016.10.034

Goossens, J. W., Van Hoesel, S., & Kroon, L. (2006). On solving multi-type railway line planning problems. *European Journal of Operational Research*, *168*(2 SPEC. ISS.), 403–424. https://doi.org/10.1016/j.ejor.2004.04.036

Goossens, J. W., van Hoesel, S., & Kroon, L. (2004). A branch-and-cut approach for solving railway line-planning problems. *Transportation Science*, *38*(3), 379–393. https://doi.org/10.1287/trsc.1030.0051

Huisman, D., Kroon, L. G., Lentink, R. M., Vromans, M. J., Huisman, D., Kroon, L. G., Lentink, R. M., & Vromans, M. J. (2005). Operations Research in passenger railway transportation. *Statistica Neerlandica*, *59*(4), 467–497. https://doi.org/10.1111/J.1467-9574.2005.00303.X

Liebchen, C. (2008). The first optimized railway timetable in practice. *Transportation Science*, *42*(4), 420–435. https://doi.org/10.1287/TRSC.1080.0240

Nachtigall, K. (1996). Periodic network optimization with different arc frequencies. *Discrete Applied Mathematics*, *69*(1-2), 1–17.

Nachtigall, K., & Voget, S. (1996). A genetic algorithm approach to periodic railway synchronization. *Computers and Operations Research*, *23*(5), 453–463. https://doi.org/10.1016/0305-0548(95)00032-1

Odijk, M. A. (1996). A constraint generation algorithm for the construction of periodic railway timetables. *Transportation Research Part B: Methodological*, *30*(6), 455–464. https://doi.org/10.1016/0191-2615(96)00005-7

Peeters, L. (2003). *Cyclic Railway Timetable Optimization* (Doctoral dissertation). https://repub.eur.nl/pub/429/

Schiewe, A., Alber, S., Schiewe, P., Schöbel, A., & Spühler, F. (2020). LinTim - Integrated Optimization in Public Transportation. https://www.lintim.net/

Schiewe, P. (2020). *Integrated Optimization in Public Transport Planning* (Vol. 160). Springer International Publishing. https://doi.org/10.1007/978-3-030-46270-3

Schmidt, M. E. (2014). *Integrating Routing Decisions in Public Transportation Problems* (Vol. 89). Springer New York. https://doi.org/10.1007/978-1-4614-9566-6

Schöbel, A. (2017). An eigenmodel for iterative line planning, timetabling and vehicle scheduling in public transportation. *Transportation Research Part C: Emerging Technologies*, *74*, 348–365. https://doi.org/10.1016/j.trc.2016.11.018

Schöbel, A., & Scholl, S. (2006). Line Planning with Minimal Traveling Time. *DROPS-IDN/660*. https://doi.org/10.4230/OASICS.ATMOS.2005.660

Serafini, P., & Ukovich, W. (1989). A Mathematical Model for Periodic Scheduling Problems. *SIAM Journal on Discrete Mathematics*, *2*(4), 550–581. https://doi.org/10.1137/0402049

Warman, M. (2014). Valuing Convenience in Public Transport. *Valuing Convenience in Public Transport*, 1–70. http://www.oecd-ilibrary.org/transport/valuing-convenience-in-public-transport_9789282107683-en

Appendix A

CPLEX Parameters

The table below shows the parameters changed in comparison with the default settings of CPLEX.

parameter	value
`IloCplex.Param.MIP.Tolerances.MIPGap`	$1e-7$

Table A.1: This table shows the non-default parameter settings for CPLEX used in this thesis.

Appendix B

Data of the test instances

B.1 OD matrix

	101	102	103	104	105	201	202	203	204	205	301	302	303	304	305	401	402	403	404	405	501	502	503	504	505
101	0	21	15	10	8	5	11	19	7	3	3	6	82	8	4	3	7	11	4	2	2	2	5	4	15
102	5	0	4	3	2	1	3	5	2	1	1	2	20	2	1	1	2	3	1	0	0	1	1	1	4
103	3	3	0	4	3	1	3	7	3	1	1	3	30	2	1	1	2	4	2	1	1	1	2	1	4
104	2	2	4	0	4	1	2	5	3	1	1	3	19	1	1	0	1	3	2	1	1	1	1	1	2
105	2	2	3	4	0	1	1	4	2	1	1	2	16	1	0	0	1	2	1	1	1	1	1	0	2
201	1	1	1	1	1	0	8	5	4	2	1	1	20	2	1	1	1	3	2	1	0	0	1	1	3
202	2	3	3	2	1	7	0	11	9	4	1	3	47	4	2	1	3	7	6	2	1	1	2	2	7
203	4	4	7	4	3	4	10	0	11	5	3	6	78	7	3	2	6	11	6	3	2	3	4	3	10
204	1	2	3	3	2	3	8	11	0	7	2	5	46	6	3	2	5	6	3	1	1	2	2	1	4
205	1	1	1	1	1	1	4	5	7	0	1	2	19	3	1	1	2	3	1	1	1	1	1	0	2
301	1	1	1	1	1	1	2	3	1	1	0	7	25	4	2	1	1	3	2	1	1	1	1	1	3
302	1	1	1	1	1	1	2	4	2	1	4	0	29	5	2	1	2	4	3	1	1	1	2	1	3
303	14	4	7	5	4	5	11	18	11	5	6	9	0	14	6	5	11	18	11	5	4	5	7	5	8
304	1	1	2	2	2	2	6	7	4	2	4	9	55	0	7	2	4	7	6	3	1	1	3	2	8
305	0	0	1	1	1	1	3	3	2	1	2	4	23	7	0	1	2	3	3	1	0	1	1	1	3
401	1	1	1	0	0	1	1	3	2	1	1	2	19	3	1	0	8	5	4	2	1	1	2	1	5
402	1	2	2	1	1	1	3	6	5	2	2	4	45	6	3	7	0	11	9	4	1	2	4	3	11
403	2	2	4	2	2	2	5	10	6	2	3	6	75	7	3	4	11	0	11	5	3	4	7	4	17
404	1	1	2	2	1	2	5	6	3	1	1	3	45	4	2	4	9	11	0	7	2	3	3	2	7
405	0	0	1	1	1	1	2	2	1	1	0	1	18	2	1	2	4	5	7	0	1	1	1	1	3
501	0	0	1	1	1	0	1	2	1	1	0	1	15	2	1	1	1	4	2	1	0	5	3	2	8
502	0	0	1	1	1	0	1	2	2	1	1	1	18	2	1	1	2	5	3	1	4	0	4	3	9
503	1	1	2	1	1	1	2	4	2	1	1	3	28	2	1	1	3	7	3	1	3	4	0	4	13
504	1	1	1	0	0	1	2	2	1	0	1	2	17	1	0	1	3	4	2	1	2	2	4	0	19
505	1	1	1	0	0	0	1	2	1	0	1	2	13	1	0	1	2	3	1	1	1	2	3	4	0

Table B.1: Origin-destination matrix of the `grid` instance.

B.2 Lines pools

Table B.2: This table shows the lines used in the 5×5 grid. The type column shows to which set of lines the line belongs. H, V, and D are the horizontal, vertical, and diagonal lines.

line number	type	#stops	stops
L1	H	5	101,102,103,104,105
L2	H	5	201,202,203,204,205
L3	H	5	301,302,303,304,305
L4	H	5	401,402,403,404,405
L5	H	5	501,502,503,504,505
L6	V	5	101,201,301,401,501
L7	V	5	102,202,302,402,502
L8	V	5	103,203,303,403,503
L9	V	5	104,204,304,404,504
L10	V	5	105,205,305,405,505
L11	D	7	201,202,203,303,403,404,405
L12	D	7	102,202,302,303,304,404,504
L13	D	7	401,402,403,303,203,204,205
L14	D	7	502,402,302,303,304,204,104

Table B.3: This table shows the lines used in the 4×4 grid. The type column shows to which set of lines the line belongs. H, V, and D are the horizontal, vertical, and diagonal lines.

line number	type	#stops	stops
L1	H	4	101,102,103,104
L2	H	4	201,202,203,204
L3	H	4	301,302,303,304
L4	H	4	401,402,403,404
L5	V	4	101,201,301,401
L6	V	4	102,202,302,402
L7	V	4	103,203,303,403
L8	V	4	104,204,304,404
L9	D	7	101,102,202,203,303,304,404
L10	D	7	101,201,202,302,303,403,404
L11	D	7	401,301,302,303,203,204,104
L12	D	7	401,402,302,303,304,204,104