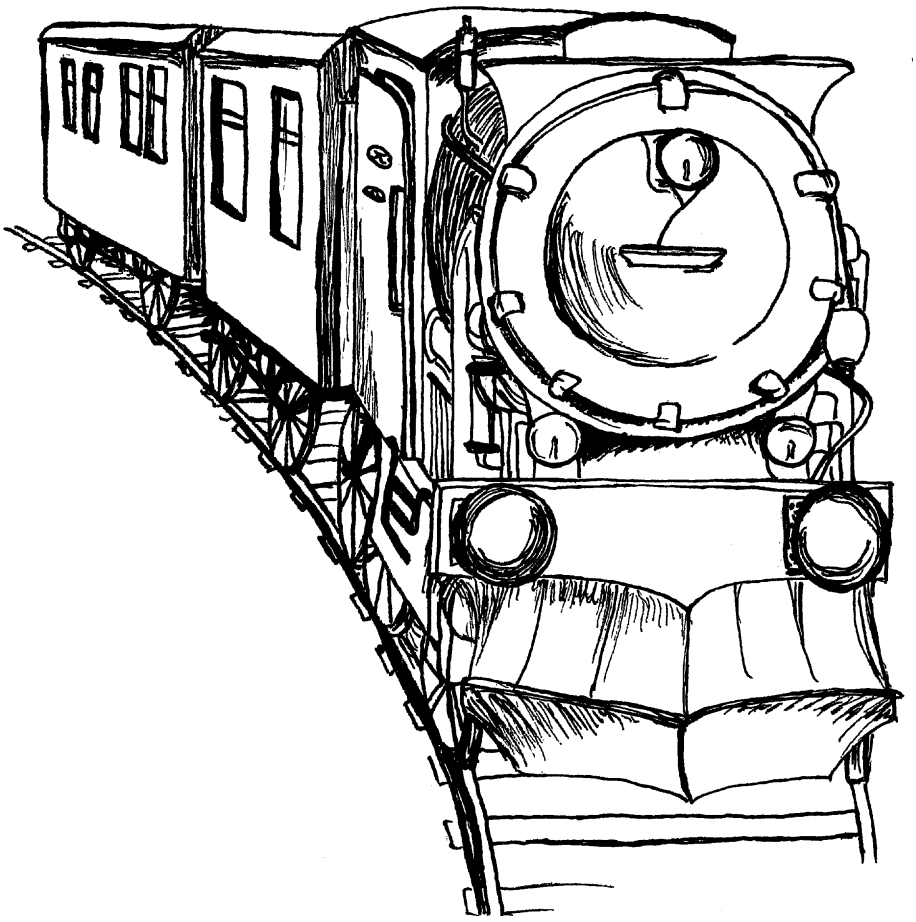


Planning a Public Transportation System with a View Towards Passengers' Convenience

by
Jonas Harbering

Planning a Public Transportation System with a View Towards Passengers' Convenience

Dissertation

zur Erlangung des mathematisch-naturwissenschaftlichen Doktorgrades
"Doctor rerum naturalium"
der Georg-August-Universität Göttingen

im Promotionsprogramm "PhD School of Mathematical Sciences" (SMS)
der Georg-August University School of Science (GAUSS)

vorgelegt von
Jonas Harbering
aus Flemhude

Göttingen, 2015

Betreuungsausschuss

Prof. Dr. Anita Schöbel, Institut für Numerische und Angewandte Mathematik, Georg-August-Universität Göttingen

Prof. Dr. Stephan Westphal, Institut für Angewandte Stochastik und Operations Research, Technische Universität Clausthal

Mitglieder der Prüfungskommision

Referentin: Prof. Dr. Anita Schöbel, Institut für Numerische und Angewandte Mathematik, Georg-August-Universität Göttingen

Koreferent: Prof. Dr. Juan Antonio Mesa, Institut für Angewandte Mathematik, Universität Sevilla

Weitere Mitglieder der Prüfungskommision

Prof. Dr. Stephan Westphal, Institut für Angewandte Stochastik und Operations Research, Technische Universität Clausthal

Prof. Dr. Andrea Krajina, Institut für Mathematische Stochastik, Georg-August-Universität Göttingen

Prof. Dr. Preda Mihailescu, Mathematisches Institut, Georg-August-Universität Göttingen

Prof. Dr. Jens Grabowski, Institut für Informatik, Georg-August-Universität Göttingen

Tag der mündlichen Prüfung: 1. Februar 2016

Contents

1. Introduction

Public transportation is increasingly used by passengers. Ever more people intend to travel on longer distances using public transportation. The consequence for operators is that they have to provide sufficient capacity in order to satisfy passengers' demand. However, public transportation operators are facing difficulties as costs are also increasing. Examples for influencing cost factors are labor wages and energy consumption. Statistics on public transportation demand and influencing cost factors are provided by [Eur15]. Finally, also political goals are embodied in this topic. As public transportation is considered to be safer and more sustainable than, e.g., private transportation, (see [Ban08]) there is a political will to support public transportation usage [OEC15].

In order to provide a guidance on how to deal with the difficulties and how to achieve efficiency goals, an intergovernmental organization, the "International Transportation Forum", consisting of 57 member countries, was created in 2006. The forum describes their aim "to foster a deeper understanding of the role of transportation in economic growth, environmental sustainability and social inclusion and to raise the public profile of transportation policy" (see [ITF15]). As such, they are regularly publishing discussion papers stating the role of current and future research areas. Two recent papers are concerned with the improvement of railways, emphasizing on crucial factors from the operators' perspective ([MBP15]) and from passengers' view ([WAC+14]).

In [MBP15] the efficiency aspect of railway operators is further analyzed, stating that key indicators are, among others, passenger transportation operators' revenue to cost ratio, infrastructure managers' revenue to cost ratio, utilization of railway infrastructure, and utilization of freight/passenger transportation operators. Putting this differently, important values for both freight and passenger transportation operators are:

- generalized costs including, e.g., marketing costs, labor costs, train operating costs, and maintenance costs of fleet and infrastructure,
- revenue from passengers or companies transporting freight or demanding freight

transport, and

- utilization of freight and passenger train capacity.

See [MBP15] for a more detailed analysis of efficiency indicators. This thesis is mainly concerned with problems arising in the area of passenger transport.

Elaborating on passengers' perception of railway services, [WAC+14] reviews the key factors of the convenience of passengers, or short passenger convenience. The elements found to compose the convenience perception of passengers are:

- Accessibility, described by access and egress time. Also the walking time during the journey is embodied in this element.
- Time spent waiting at any stage of the journey.
- Availability, firstly in terms of the frequency in the timetable, i.e., the number of planned services. Secondly, in terms of the timetable due to delays, i.e., the number of actual services.
- The number of transfers. This inconvenience element is meant to be in addition to the time spent waiting.
- Variability in the traveling time due to unforeseen reasons.
- The amount and quality of information provided.
- Crowding due to overloaded trains.

For a more detailed analysis of passenger convenience elements, see [WAC+14].

These two position papers summarize very extensively the two main contradicting goals of railways operations, namely, reducing operators' costs and improving passengers' convenience. In this thesis different aspects of these contradicting views are analyzed in the presented publications. Mainly, passenger convenience goals are addressed without losing sight of the operators' perspective.

- In [CHS14] (see Addendum A) an estimate for passengers' traveling time is minimized and an acceptable distance to access the rail system for the passengers is ensured. Two integer programming formulations are proposed and, together with an additional integer program from literature, their performances are compared. In this work passengers' traveling time as well as access and egress time are considered. Thus, this work mainly addresses passenger convenience. The summary of this work is given in Section 3.1.

- In [GHS15a] (see Addendum B) line pools, composing sets of potential lines, are generated with a focus on generalized costs and a simultaneous view on passenger transfers. An algorithm solving the arising model is proposed and various influential parameters are analyzed. This work can be classified to consider both operator costs and passenger convenience. It is summarized in Section 3.2.

- The objective of [Har15] (see Addendum C) is to find lines which minimize passenger transfers. Thereby, it is intended to show that the resulting lines serve well in order to combat delays. In order to ensure that not too many lines are used, we bound the number of trains on each edge. The resulting model is formulated as an integer program and a heuristic algorithm is proposed to solve the program. This work mainly addresses passenger convenience in two aspects, which are passenger transfers and availability, but also regards the costs. The summary of this work can be found in Section 3.3.

- The work [HRS15] (see Addendum D) focuses on the utilization of railway infrastructure. It is motivated by the situation in India, in which trains are highly overloaded and using the infrastructure more efficiently is striven for. This work states a model for the optimal utilization of the infrastructure and is concerned with the complexity of the problem. The utilization of the infrastructure is, in the first place, one of the major cost aspects. However, also important elements of passengers' convenience are addressed. Hence, this work emphasizes on both costs and passenger convenience perspectives. In Section 3.4 this work is summarized.

- In comparison to the previous works, the focus of [MHS$^+$15] (see Addendum E) is different. The aim in this work is to help operators understand the propagation of delays better and thus to enable railway operators to combat the spreading of delays more effectively. In order to achieve this, algorithms are proposed deducing the sources of delays. Hence, it is both related to generalized costs from operators' perspective and delays from passengers' perspective. The summary of this work can be found in Section 3.5.

The above analysis of the contradicting goals in planning public transportation systems and the contributions by the different publications show the merit of this thesis, which is to provide deeper insight into passenger convenience aspects without losing sight of cost related issues.

In addition to the modeled aspects of operators' and customers' perspective, works within the planning process of public transportation systems can be classified by the

methods used. It means that not only the constraints and objectives are important but also the mathematical methods applied.

In the following Section 2 a literature overview stating general ways of approaching the planning of public transportation systems is given. The two contradicting goals – operators' and passengers' perspective – are highlighted throughout the overview and the contributed works are embedded in a broader literature context. After that, the publications of this thesis will be summarized in Section 3. The discussion of the results is given in Section 4, followed by the final conclusions in Section 5. Section 6 gives an overview of the contributions of J. Harbering to the publications of this thesis.

2. General Literature Overview

The planning of public transportation systems, i.e., bus systems operating on the streets and railway systems operating on tracks, is composed of many planning elements. The following is mainly concerned with railway systems but almost all planning phases appear for bus systems in the same way.

The first optimization based research about public transportation aspects is found in [Pat25]. Thereafter, literature within this area is scarce and only in the 70's and 80's the topic of public transportation planning begins to attract attention within the optimization community. Early literature reviews on the topic are given by [CW86, BWZ97]. In these works the contradicting goals of minimizing costs and serving passenger demand are already highlighted. Additionally, both works present the following common approach. The public transportation planning is decomposed into a sequence of mathematical optimization problems. In [CW86] the problems identified are Network Design, Setting Frequencies, Timetable Development, Bus Scheduling, and Driver Scheduling. [BWZ97] states the problems slightly different, namely, Network Planning, Line Planning, Train Schedule Generation, Scheduling: Rolling Stock and Personnel, and Real Time Traffic. In the following years, many other problems and specifications of the former have been considered. However, the problems highlighted in Figure 1 have been accepted as composing a part of the main sequence of problems within the planning of public transportation ([HKLV05, DH07, LM07, LLER11, Sch12]).

In the *network design* problem an infrastructure network consisting of nodes and edges, i.e., stops and tracks, is to be designed. The *line planning* problem takes the infrastructure network as an input and provides lines with frequencies on which trains are riding. The lines are given as paths in the infrastructure network. In the *timetabling* problem arrival and departure times for all trains at all stations are computed. For the *delay management* problem a set of delays in the operated timetable is given. A disposition timetable is computed in order to minimize the impact for passengers.

The advantage of decomposing the sequence into the stated problems is twofold. First,

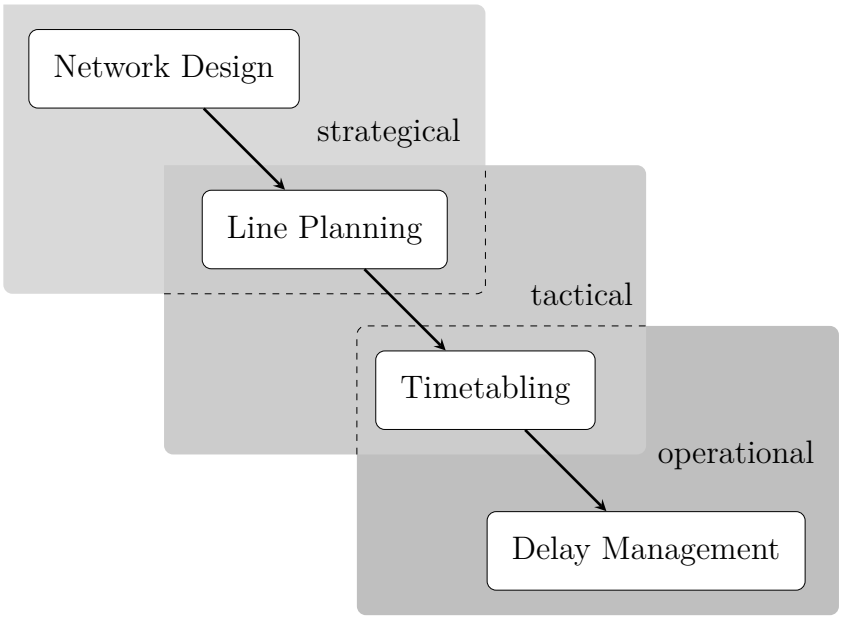


Figure 1: A part of the planning stages in public transportation

planning all elements at the same time is by far too complex. A mathematical model could maybe be stated but it would be almost impossible to compute feasible or optimal solutions. Additionally, deriving any theoretical results for such a model would be very difficult. Hence, a decomposition seems to be necessary in order to be able to study the system planning. The other reason why the planning is decomposed into these problems is due to practical considerations. The planning of public transportation involves different aspects which are to be planned with different planning horizons. The horizons are of strategical, tactical, and operational nature (see e.g. [LLER11]). In particular, the network design is considered at the strategical level ([BWZ97, DH07, LLER11]). The line planning is in some works allocated to the strategical level ([HKLV05]) and in others to the tactical level ([BWZ97, DH07, LLER11]). The timetabling problem is usually attributed to either the tactical ([BWZ97, HKLV05, LLER11]) or the operational level ([HKLV05, DH07]). Finally, all reviews agree that the delay management problem occurs at the operational level, sometimes more specifically on a day-to-day level. The view adopted here is highlighted in Figure 1 by the shadings while other possible problem assignments are indicated by the dashed lines.

The fact that some of the problems cannot be clearly assigned to only one planning level shows that different aspects of the particular problem can be studied. Depending on

the precise aspect, such a problem can be allocated to either of the two planning levels.

Decomposing the public transportation planning process into such a sequence has the above mentioned advantages, but from an optimization point of view there is a clear disadvantage. Using the output of one planning problem for the input of the next planning problem possibly restricts solutions, and hence it is very unlikely to obtain system-wide optimal solutions.

Note that passenger routing problems have not been mentioned within the above stated planning sequence. The assignment of passengers to their planned and actual paths can be considered within each of the planning phases and is thus not considered to be a problem of the sequence. Planning the passenger paths within a planning phase can be treated in two ways. Considering the assignment of passenger paths separately from the planning problem raises a chicken-egg dilemma. For example, at the stage of line planning, the passenger paths depend on the operated lines while the lines can be chosen according to passengers' wishes. A workaround is to assign the passengers to paths at the same time as solving the planning problem. In general, this turns out to be very hard (see [Sch14]).

After many approaches which decompose the planning of public transportation systems into a sequence have been considered in the literature, this idea is being gradually reverted. The idea of integrating planning aspects of different problems, so-called integrated public transportation planning, is receiving more and more attraction [HDD01, Hui04, Lie08, MS09, GSS13, MOP14]. To this end, also the software package LinTim (see [GHS15c]) has been developed. The aim of this software tool is to analyze the public transportation planning problems. To this end, generic interfaces between the problems are developed. This enables the user to study dependencies and correlations between the different planning problems. An overview together with an exemplary analysis based on LinTim is given by [GSS13]. Computations within this thesis are mainly based on data and algorithms from LinTim.

This thesis comprises five publications covering aspects of all four consecutive planning problems of Figure 1 with the main focus on improving public transportation with respect to the convenience perception of passengers, as outlined in the introduction. The most important literature concerning the planning problems is discussed in the following. It includes the literature on which the publications of this thesis are based.

2.1 Network Design

In this section the network design problem is presented and some of its facets are discussed. In particular, literature on which [CHS14] (see Addendum A) is based, is

highlighted. The summary of this work is given in Section 3.1.

In the network design problem the infrastructure network is planned. In its most basic version only population data containing the location and the size of population areas is given. Based on this, infrastructure elements, composing the public transportation network (PTN), are planned. The public transportation network is a graph, consisting of nodes and edges, representing stops and streets or railway tracks, depending on the public transportation system. Hence, in this problem it is decided where to locate nodes and which stops to connect by edges. The objectives usually represent operator interests as it is assumed to be a problem mainly concerned with costs. Literature reviews are given in [LMO00, LMOP11].

In [LJMO02, MW03, HLS$^+$01, MSW06, SHLW09] a previously existing network together with information about demand nodes in the plane, e.g., cities, is assumed to be given. Based on this existing network and the demand nodes, stops are added, and thus the network is extended by new infrastructure elements. In these problems the location of the stops is sought, hence it is called stop location. In [LJMO02, MW03] the objective is to maximize the coverage due to new stations, i.e., only focusing on passenger interests.

In contrast, the objective in [MSW06, SHLW09] is to minimize the number of additional stops. Assuming the traveling speed to be constant, the minimization of the number of stops to be built can be viewed as the minimization of a passengers' traveling time estimate. Additionally, the passenger convenience element is reflected by ensuring that the distance between each demand node and the closest stop in the infrastructure network does not exceed a certain distance. By assigning a cost to each stop, the minimization of the number of new stops can also be viewed as the cost-related operators' view. The two aspects of costs and passenger convenience are represented in [Sch05] in which a bicriteria model is formulated. In this work the number of stops is minimized and the number of people reaching the infrastructure network within a certain distance is maximized. In [HLS$^+$01] the locations of new stops are judged by the difference in travel times. The difference results from time savings due to new stops which are closer to demand nodes and from travel time increases for passengers which are traveling in the network and endure additional halting times.

Based on [MSW06, SHLW09], the stop location problem is extended in [CHS14] (see Addendum A) with a more realistic traveling time function. The approach chosen in [CHS14] introduces the realistic traveling time function by precisely modeling vehicles' acceleration and deceleration phases. Such a traveling time function, derived in [Vuc81], has been considered in a similar context in [DDW09].

The approach in [CHS14] mainly focuses on the passengers, with the aim to minimize a passengers' traveling time estimate subject to the constraint that all passengers demanding access to the infrastructure network receive a station within a certain acceptable distance.

The result of the network design is a public transportation network (PTN) which is the input for the subsequent line planning problem.

2.2 Line Planning

This section is devoted to giving a broad overview of line planning models. The two works [GHS15a] and [Har15] (see Addendum B and Addendum C, respectively) fall into this area. Their summaries are given in Section 3.2 and 3.3, respectively.

Reviews of the line planning problem are given by [BWZ97, LLER11, Sch12]. In this phase it is assumed that a public transportation network (PTN), consisting of nodes and edges, i.e., stops and tracks, is given, and lines, represented by paths on this network, are sought. Each line can be operated with a frequency which is also to be determined. For the sum of the frequencies, lower and upper bounds on every edge are given. While the lower frequency bounds specify the least number of trains that have to pass each edge, the upper frequency bound states the maximum number of trains per edge. By requiring the minimal frequency, a least amount of service is ensured. The upper frequency bounds reflect security and noise reasons. In some works, additionally, a budget constraint has to be ensured. For this, a cost parameter is associated to each line. The cost parameter can, e.g., be proportional to the length of the line plus a fixed cost term. The overall costs, bounded by the budget, are computed as the frequency of a line times its costs, summarized over all lines. Additionally, in some works also passenger data is assumed to be given in the form of origin-destination (OD) data. One entry in this data specifies how many passengers are interested in using public transportation from a particular origin to a particular destination. Origin and destination are thereby usually stops in the PTN. The aim is to find a set of lines together with frequencies such that lower and upper frequency bounds and the passenger demand are satisfied and the budget is not exceeded. The solution, consisting of the set of lines together with frequencies, is called a line concept.

Note that in most presented works from the network design problem the available data states demand points, whereas in the line planning many models consider origin-destination information as input. However, approaches applying the gravity model can be used to obtain traffic count data or an origin-destination matrix [Voo13]. The traffic count data can be used to compute the lower frequency bounds for all edges (see [Sch12]).

See [BGP08, Sch12] for a more detailed overview of elements of line planning and an

examination of the history of the development of line planning models.

In the literature this problem is considered in two steps. In a first step a set of candidate lines, the line pool, is computed and used as input to the second step problem. The first problem is called the *line pool generation* problem. The problem in the second step chooses a subset of lines from the line pool and assigns frequencies to them. This problem is called *frequency setting problem*.

Only few approaches for the line pool generation problem exist, most of which rely on simple heuristics generating lines based on shortest paths [SBP74, Man80]. An important difference between the intention of other approaches similar to the line pool generation problem and the one considered in this thesis is the following. In other approaches the basic idea is to generate only lines to which frequencies are assigned in the second step problem [KK09, FM10].

In contrast, the line pool generation problem considered in this thesis has the intention to provide a line pool which is not too small so as to provide flexibility for the subsequent frequency setting problem. However, the line pool should also not be too big in order to ensure computability for the frequency setting problem. The work [GHS15a] (see Addendum B) provides such a set of potential lines and the approach, inspecting influential parameters for good line pools, has not been considered before. The result from [GHS15a] is a line pool which is passed to the second step problem, i.e., the frequency setting problem, also usually referred to as the line planning problem. As described, in such frequency setting problems a predefined set of candidate lines is assumed to be given and frequencies for the candidate lines are sought.

The most basic version of a frequency setting problem with a given line pool is the cost model. In this model the only constraints are the bounds on the sum of the frequencies for each edge and the objective is to minimize the costs as introduced by [CvDZ98]. Adopting a passenger view, [BKZ97] include the constraints on the frequencies and propose to model direct passengers, i.e., passengers who do not have to transfer. For each passenger, i.e., od-pair, a preferable path in the infrastructure network is computed. Preferable refers to the preferences of the passenger. It is proposed to compute the shortest path as the preferable path. Then a passenger is considered to travel without a transfer if its preferable path lies entirely within an operated line. With this model the number of direct travelers is maximized. A different approach minimizing the passenger traveling time, has then been proposed by [SS06]. All these works consider a line pool to be given as input.

In contrast to the previous version of breaking the line planning problem into two subsequent planning problems, there also exist works which consider the line pool generation

problem at the same time as the frequency setting problem. A very promising example of this is [BGP07]. In accordance with the literature, the frequency setting problem is in the following referred to as the line planning problem.

Similar to [BGP07], the model presented in [Har15] (see Addendum C) minimizes the number of transfers. Also the authors of [SS06] show that their model can be adapted to minimize the number of transfers. [BGP07] and [SS06] model that passengers can be assigned to any path in the network. Due to capacity restrictions some passengers might need to endure long detours. Restricting the length or the traveling time of the paths would significantly increase the complexity, as noted in [BGP07]. In contrast, in [Har15] the passengers are only allowed to travel on fixed paths in the infrastructure network. These fixed paths are considered as preferable to the passengers. With this model the passenger routing is included in the line planning problem and the length of the passenger paths is restricted without increasing the complexity of the problem. The solution method proposed for solving the proposed problem is column generation, similar to [BGP07]. Another important difference between [BGP07] and [Har15] is that the former considers the line pool generation problem at the same time as the line planning problem. The latter considers the line pool to be given as an input.

Furthermore, the aim in [Har15] is to show that minimizing the passenger transfers has a positive effect on combating the spreading of delays. This aim is supported by an analysis of different line concepts towards the applicability concerning the spreading of delays.

In terms of the contradicting goals stated in the introduction, the work [Har15] can be mainly viewed as adopting a passenger view by minimizing their transfers. Due to the upper frequency bounds on the edges, frequencies are not allowed to be arbitrarily high and thus costs are also implicitly controlled. Finally, the result of a good performance under delays serves both the passengers, as they have to endure less deviation from the scheduled plan, and the operator, as costs decrease with smoother operations.

The result of the line planning is a set of lines together with frequencies which is the input for the timetabling problem.

2.3 Timetabling

In this section an overview on relevant literature in the area of timetabling is given. Furthermore, relevant literature in relation to the problem considered in [HRSS15] (see Addendum D) is highlighted. The summary of [HRSS15] is given in Section 3.4.

For the timetabling problem a set of lines with associated frequencies is given. In some works also passenger paths are considered as input. Then, for all edges, lower and

upper bounds on the drivings and haltings of all trains are specified as well as lower and upper durations for passenger transfers. Additionally, track and station capacities can be considered. If they are considered, minimal time distances between trains on track segments and in stations have to be enforced. Based on this input data, a feasible timetable assigns arrival and departure times to all trains at all stations. The timetable is feasible if the lower and upper bounds on the traveling times, on the passenger transfer durations, and on the distances between trains are satisfied. This general timetabling problem can be expressed on a graph which is called the event-activity network, see [SU89, Nac98].

This general formulation of the timetabling problem is still a broad formulation, which can be divided in two distinct variants. Models serving the passenger interest of periodicity form the first variant. By requiring periodicity, a timetable is sought which repeats after a given period of usually one or two hours. [SU89] are the first to introduce this periodic model. General overviews on different methods solving the periodic timetabling problem are given by [CFT02, Lie07, LM07, CT12, LLER11, SG13].

The second variant is the aperiodic case in which no further requirement concerning the periodicity is made. For a model of the aperiodic timetabling case, see, e.g., [CCT10, LLER11]. For results related to other scheduling problems appearing in the timetabling phase, it is referred to [dBBB$^+$13] (crew scheduling) and [Mar06] (vehicle scheduling). For other scheduling problems, such as maintenance scheduling and shunting, literature survey papers are [CTV98, GMMW09].

Some of the works summarized in [LLER11] consider the aperiodic timetabling on a single bidirectional graph. The basic timetabling problem on a single line can be transferred to a machine scheduling problem in the following ways. The block segments can be represented as machines and the trains traversing the bidirectional line are represented as jobs [CS08]. Each train has a specific traveling time on each block, i.e., each job has a specific processing time on each machine. Results on this problem within the machine scheduling context are given by [BBB77, Dus88]. In these problems it is further assumed that jobs start at both end points of the sequence of machines and have to be processed on every machine in turn. The objective is to minimize the makespan. Summaries on related results are given in [Sev94, CPW99]. In [HRSS15] (see Addendum D) the setting is simplified by assuming that the processing time on a machine is equal for all jobs. In terms of the trains it means that the traveling times on a block is the same for all trains. The complexity of the resulting problem is analyzed further narrowing down the gap between polynomially solvable and NP-hard instances.

The problem in Section 3.4 ([HRSS15]) has the aim to make the best use of the provided

track capacity by scheduling as many trains as possible. Increasing the capacity usage is one of the key indicators of operators' performance and hence this problem has a focus on operators' efficiency. However, by assuming that there are sufficient people demanding transport, the results of this problem also provide the highest possible availability. Possible crowding effects of passengers are counteracted by providing as many trains as possible. Hence, in accordance with the contradicting goals outlined in the introduction, this problem provides schedules which are both desirable in terms of operators' cost elements and in terms of passenger convenience elements.

The result of the timetabling problem is a feasible timetable. This timetable is then the input for the subsequent delay management problem.

2.4 Delay Management

In this section an overview of the basic delay management problem and relevant literature is given. Since the problem considered in [MHS+15] is related to the delay management problem, relevant literature in relation to this work is also discussed here. The summary of [MHS+15] is given in Section 3.5.

When operating a timetable of a public transportation system, given from the previous planning phase, delays are unavoidable since they are due to exterior reasons. Possible causes for delays are construction sites improving the infrastructure or adjacent constructions, bad weather conditions (e.g., lightnings damaging the infrastructure), demonstrations, and malfunctioning of particular important infrastructure elements. Such events introduce delays into the timetable. These primary delays are so-called source delays. Due to various mechanisms these source delays then spread within the system inflicting delays on other trains (see [Sch01, SS10]). These spread or inflicted delays are so-called propagated delays since they only arise due to the primarily introduced source delays. The propagated delays occur due to various dependencies within the transportation system. Naturally, the particular train which received the source delay in the first place is only able to catch up on the delay by time margins in the timetable. In case the time margin, or so-called buffer, is smaller than the initial delay the train takes the source delay to the stations it passes.

There are also other dependencies, which promote the spreading of delays. The main dependencies can be grouped into two sets. At the stations, passengers usually have planned transfers. If passengers wanted to transfer at a particular station from a delayed train to a connecting train which departs on time, the passengers would have to wait for their next connection. Otherwise, if the connecting train waits for the delayed transferring

passengers, the delay carries over to the next train in that station. Inter-train dependencies of the type where passengers transfer are called *wait-depart decisions*. After a train leaves a station, it enters a block section which is allocated only to this particular train for a given time slot. Hence, each block has a sequence of trains assigned to it. If one train within the sequence gets delayed, the sequence might have to be adapted. The decision related to whether to maintain or to reorder this sequence is called *priority decision*. The aim of delay management is to prevent spreading in terms of minimizing the delay of the passengers at their end station by taking wait-depart and priority decisions.

A theoretical analysis of the spreading of delays can be found in [Gov98, KK15]. A basic variant of the delay management problem can be found in [Sch01, SS10]. Note that these works are considered to be offline, i.e., the delay information is assumed to be given when planning the new disposition timetable. Conversely, literature considering the online or real-time version of this problem can be found in [BHLS10, MOP13, BBK13, CHK$^+$14]. Deciding about how to deal with delays once they occur is not the only approach. There exist measures even before the delays occur. A widely discussed method is to add buffer times in the timetable making it thereby more robust against delays. Different concepts are discussed in [LSS$^+$10] and a literature review on related works is given in [CT12]. Other delay management aspects such as crew and vehicle rescheduling are reviewed in [JGPC$^+$09].

Once the delays have occurred and have been propagated it is not an easy task to determine the source of the delays ([YKAT13]). A method proposed for detecting the origin of a spreading process is given in [BH13, MKS$^+$14]. In these works a network based method has been applied to the disease spreading phenomena. This method is adopted in [MHS$^+$15] (see Addendum E) to deduce the origin of the delay spreading by modeling the spreading as a stochastic process on the railway infrastructure network.

This work can be viewed as to provide operators with a more sophisticated understanding of the spreading of delays. It can be used to analyze real time spreadings and combat them even more effectively. If operators have a deeper understanding of the spreading of delays, passengers benefit from this in the long term. Hence, eventually mainly passengers benefit from this work.

In this literature overview the stepwise solution approach of public transportation planning has been highlighted. The contradicting goals of operators' and passengers' perspectives have also been discussed. The next section is devoted to summarizing the papers contributing to this thesis.

3. Paper Summaries

This section summarizes the papers representing the different parts of this thesis. The order in which the summaries are presented is the same as in the sequence of planning stages of public transportation planning, as outlined in Section 2 (see Figure 1).

First, Section 3.1 contains the summary of two publications. The two publications are [CHS13] and [CHS14] (see Addendum A), where [CHS14] is an extended version of [CHS13], and hence in the following it is referred to [CHS14]. Within the network design phase, a stop location problem is considered in this paper. The authors intend to model the vehicles' traveling time function realistically by taking into account acceleration and deceleration. The new objective function and resulting modeling aspects are analyzed.

The next paper summary is given in Section 3.2 where [GHS15a] and [GHS15b] are summarized. Again, [GHS15b] is an extension of [GHS15a] and in the following it is referred to [GHS15b] (see Addendum B). This paper considers a problem connecting the network design phase and the line planning phase. The line pool generation problem extracts a set of possible lines from the set of all paths on a network. This new set is then called line pool and used as an input for the line planning step. The task is to generate a sufficiently big set such that subsequent line planning problems have a large degree of freedom for choosing a solution. In contrast, the line pool shall be as small as possible such that the subsequent line planning models can be executed efficiently.

In Section 3.3 the paper [Har15] (see Addendum C) is summarized. A modeling approach is presented in which lines are chosen such that the total number of transfers is minimized. The passengers paths are contained as decision variables in the integer program and an efficient algorithm is proposed solving the arising problem. Even more, analyzing the resulting line concepts, the lines computed with this model show a positive effect on the delay resistance of the public transportation system.

Subsequently, the publication [HRSS15], based on [HRS15], is summarized in Section 3.4 (see Addendum D). As [HRSS15] is the more extended version, this work is referred to in the following. Given is a linear graph with trains on both ends which are to traverse

the graph in opposing directions. The only place to pass each other is at the intermediate stations where an unlimited capacity is assumed. The objective of the problem is to minimize the makespan, i.e., the time between the first departure of the first train until the last arrival of the last train. In this work the authors are interested in the complexity of various specifications of this problem.

Finally, Section 3.5 gives a summary on the paper [MHS+15] (see Addendum E), which is based on [MHS+14]. Given a planned timetable, unpreventable delays propagate within the system. Then, given a spreading pattern, deducing the origin of the spreading turns out not to be an easy task. In this work the authors propose methods, which partly also proved successful in other areas of application, and discuss their applicability on various test settings and real data sets.

3.1 Minimizing the passengers' traveling time in the stop location problem

by E. Carrizosa, J. Harbering, A. Schöbel

Published at ATMOS 2013 Proceedings, 14 *pages paper*

Revision submitted to the Journal of Operational Research Society, 24 *pages paper*

As outlined in the literature overview in Section 2.1, the stop location problem is a problem within the network design phase. Given a previously existing infrastructure network and demand points (e.g., cities), the aim is to extend the existing network by introducing new nodes such that all passengers have access to the infrastructure within a given range of distance. In earlier publications the objective is to minimize the number of new stations (see [MSW06, SHLW09]). With this minimization it can be argued that, assuming fixed halting times of the trains at stations and constant cruising speed while driving, also a vehicles' traveling time is minimized.

In [CHS14] the authors' intention is to go beyond this very approximate modeling and model vehicles' and passengers' travel times more realistically. The aim is to show that there is a significant difference considering realistic traveling time in comparison to former modeling ideas. As outlined in the literature overview in Section 2.1, stop location models have so far only been considered assuming a fixed cruising speed for trains. This was assumed to properly model the trains movement and give a sufficiently accurate picture on the decision where to locate new stations. If, in contrast, the movement of trains is modeled considering accelerating and deceleration, the objective function and the solutions

are significantly different. Based on this, the authors of [CHS14] introduce a traveling time function depending on the acceleration a_0, the deceleration b_0 and the cruising speed v_0.

Lemma 1.1 *([KWH75, Vuc81]) Let a maximum cruising speed $v_0 > 0$, an acceleration of $a_0 > 0$ and a deceleration of $b_0 > 0$ of a vehicle be given. Then the vehicles' driving time for a distance d between two consecutive stops is given as*

$$T(d) = \begin{cases} \sqrt{\frac{2(a_0+b_0)}{a_0 b_0} d} & \text{if } d \leq d^{max}_{v_0,a_0,b_0} \\ \frac{d}{v_0} + \frac{v_0}{2a_0} + \frac{v_0}{2b_0} & \text{if } d \geq d^{max}_{v_0,a_0,b_0} \end{cases} \quad \text{where} \quad d^{max}_{v_0,a_0,b_0} = \frac{v_0^2}{2a_0} + \frac{v_0^2}{2b_0}$$

This traveling time function has already been introduced and analyzed in [DDW09, Vuc81]. The following function properties can be shown.

Lemma 1.2 *([DDW09])*
$T(d)$ is continuous, differentiable, concave, subadditive and monotonically increasing. Furthermore, for any d we have $\sqrt{\frac{2(a_0+b_0)}{a_0 b_0} d} \leq \frac{d}{v_0} + \frac{v_0}{2a_0} + \frac{v_0}{2b_0}$.

The function $T(d)$ describes the accurate computation of the driving time of vehicles. In addition to that, the authors of [CHS14] further elaborate on this function as part of a realistic passengers' traveling time estimate function.

Therefore, let $G = (V, E)$ be a previously existing graph consisting of stops V and edges E. Let, furthermore, $\mathcal{S}$ be the set of all possible locations for new stops. Understanding $e \in E$ as the set of all points on e, $\mathcal{S}$ is given as $\mathcal{S} = \bigcup_{e \in E} e$, and let $\mathcal{S}_e = \mathcal{S} \cap e$ for all $e \in E$. For a given set $S \subset \mathcal{S}$ and $e \in E$, S_e is defined as the set of stops on e, i.e., $S_e = S \cap e$. Furthermore, $E'(S)$ is defined as the set of new edges obtained by adding the nodes S as stops. Note that all stops from V are fixed and considered to be built. By adding the set of stops S to the previously existing network, the new network $G' = (V(S), E'(S))$ with $V(S) = V \cup S$ is obtained. An example for constructing a new network based on a previously existing network is depicted in Figure 1 and Figure 2.

In order to account for passengers' traveling time on the network, let also weights w_e for all $e \in E$ and a general halting time t be given. Then, the overall passengers' traveling time function in dependence of the finite set of stations S to be built is stated as

$$g(S) = \sum_{e \in E} w_e |S_e| t + \sum_{e \in E'(S)} w_e T(d_e), \tag{1}$$

where d_e is the length of edge $e \in E'(S)$.

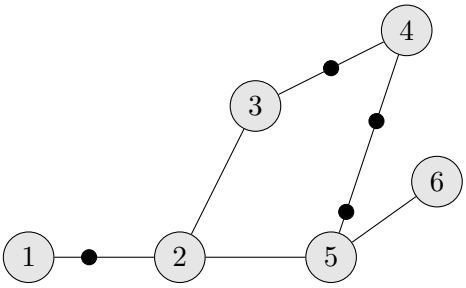


Figure 1: Locating 4 new stops on a network $G = (V, E)$.

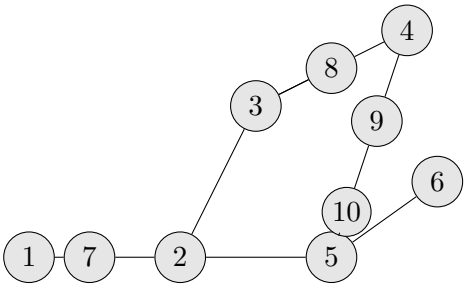


Figure 2: The new network $G' = (V \cup S, E'(S))$ with 4 new stops.

The aim of the stop location problem, studied in [CHS14], is to minimize this traveling time while ensuring that all passengers can reach the network of transportation. Passengers in a given demand point in the plane can reach the transportation network if the distance between the demand point and the next station is less than a given threshold.

More formally, assume a finite set $\mathscr{P} \subset \mathbb{R}^2$ of demand points and the network G, embedded in the plane, to be given. Based on a distance function *dist* derived a from norm and a radius $r > 0$, it is defined that a demand point $p \in \mathscr{P}$ is covered by some set $S \subset \mathcal{S}$ if $dist(p, s) \leq r$, for some $s \in S$. The cover of a set of locations S is then given as

$$cover(S) = \{p \in \mathscr{P} \ : \ dist(p, s) \leq r \text{ for some } s \in S\}.$$

Finally, assembling the different modeling parts the following problem is obtained.

(**SL***) Let $G = (V, E)$ be a graph, $\mathscr{P} \subset \mathbb{R}^2$ be a finite set of points, and $v_0 > 0$, $a_0 > 0$ and $b_0 > 0$ be the parameters for the passengers' traveling time function g in (1). Find a subset $S^* \subset \mathcal{S}$, such that $cover(S^*) = \mathscr{P}$ and $g(S^*)$ is minimized.

The first result makes an assessment about the complexity of (**SL***). The proof is done by a reduction from the NP-hard stop location problem (**SL**) from [SHLW09], in that work called (**CSLP**).

Theorem 1.3 *(**SL***) is NP-hard.*

Based on the concavity of the objective function, the authors of [CHS14] are able to deduce a finite dominating set of possible stop locations in the following.

For edge $e = (i, j) \in E$ define the set $T^e(p)$ as $T^e(p) = \{s \in e : dist(p, s) \leq r\}$. The following lemma proves the shape of the set $T^e(p)$ for an $e \in E$ and a $p \in \mathscr{P}$.

Lemma 1.4 *([SHLW09]) For each demand point $p \in \mathcal{P} \subset \mathbb{R}^2$ the set $T^e(p)$ is either empty or an interval contained in edge e.*

Having the set $T^e(p)$ as an interval, the respective end points can be used further. Let f_p^e and l_p^e be the lower and upper bounds of the interval $T^e(p)$, if they exist, then they form a finite set of candidate stops:

$$\mathcal{S}_{cand} = \bigcup_{e=(i,j)\in E} \left(\bigcup_{p\in\mathcal{P}} \{f_p^e, l_p^e\} \cup \{i, j\} \right)$$

Finally, the authors of [CHS14] show that the set $\mathcal{S}_{cand}$ is indeed a finite dominating set.

Theorem 1.5 *([CHS14]) Either (**SL***) *is infeasible, or there exists an optimal solution $S^* \subseteq \mathcal{S}_{cand}$.*

This theorem means, that instead of using the continuous set $\mathcal{S}$ as a set of possible locations, it is sufficient to consider locations from the set $\mathcal{S}_{cand}$, which is finite. The authors use this result in order to model the problem as an IP. More precisely, two IP formulations are found and their computational performances are compared. Both IP formulations use variables for the candidate stops and all edges that could possibly result under any combination of candidate stops. The two formulations differ only in the way the candidate stop variables and the edge variables are related.

The condition to be modeled ensures that an edge is constructed if and only if the two end stops are built and no intermediate stop is built. The first formulation (IP-SL*) models the relating condition in a straight-forward fashion, requiring this condition to be true for every possible edge. The second formulation (IP-2-SL*) uses flow-like constraints, requiring that each newly constructed stop must have exactly two constructed, adjacent edges.

Both formulations together with the standard stop location IP formulation (IP-SL), studied in [SHLW09], have been implemented and solved with Xpress, using LinTim. From all the results only the following are discussed here.

- In the experiments the additional realistic traveling is evaluated for all solutions. The term additional refers to the amount of time which is added due to the new stops. The decrease in additional realistic traveling time between (IP-SL) and (IP-SL*) or (IP-2-SL*) is significant ($2.1\% - 16.6\%$) on small but realistic instances. On

the tested instances the number of stations built does not vary among the different models, hence the minimal number of stops is always constructed.

- (IP-2-SL*) performs better than (IP-SL*) in terms of computation time. This result shows that the flow-like formulation (IP-2-SL*) can be solved faster than (IP-SL*). Hence, for further computations and extensions the IP formulation (IP-2-SL*) seems more promising.

As described above, the main focus of this work is to decrease the passengers' traveling time estimate. Additionally, it is ensured that all passengers who demand access to the infrastructure network can reach the network. Hence, the access and egress time, mentioned as important elements of passenger convenience in the introduction, receive a special focus.

3.2 Generation of Line Pools

by P. Gattermann, J. Harbering, A. Schöbel
Published at CASPT 2015 Proceedings, 17 *pages paper*
Submitted to the journal Public Transport, 20 *pages paper*

This problem does not appear in the classical sequence of problems in public transportation planning but is still a key problem connecting the network design and the line planning phases. As outlined in the literature overview in Section 2.2, a set of possible lines, the line pool, is given as input in the line planning phase. Since a line is a path in a graph, this line pool is a subset of the set of all possible paths in a graph. If, on the one hand, the resulting line pool is too small, only few possible combinations allow a feasible line concept. On the other hand, if the line pool is very large, the line planning problems cannot be solved efficiently. Hence, the aim of this work is to design an algorithm delivering line pools which guarantee low computation times in the line planning phase but which also allow a high degree of freedom for finding a line concept.

From the network design phase, e.g., the previously summarized work, a public transportation network (PTN) $G = (V, E)$ and demand nodes are given. The demand nodes can be used to compute origin-destination (OD) data with a gravity model approach (see [Voo13]). From this lower (f_e^{min}) and upper (f_e^{max}) frequency bounds are obtained for all edges $e \in E$. Given a line pool $\mathscr{L}$, the common constraint in most line planning models, which determine line frequencies f_l for all $l \in \mathscr{L}$, is based on the upper and lower frequency

bounds, i.e.,

$$f_e^{min} \leq \sum_{\substack{l \in \mathscr{L} \\ e \in l}} f_l \quad \text{for all } e \in E, \tag{2}$$

$$\sum_{\substack{l \in \mathscr{L} \\ e \in l}} f_l \leq f_e^{min} \quad \text{for all } e \in E. \tag{3}$$

The lower frequency bounds usually reflect traffic loads. The upper frequency bounds resemble noise and security measures. With given costs $cost_l$ for all lines $l \in \mathscr{L}$ the (**Cost**) model adopts constraints (2) and (3) with the additional objective of minimizing the costs:

$$(\textbf{Cost}) \qquad \left\{ \min \sum_{l \in \mathscr{L}} f_l cost_l \ : \ (2) \text{ and } (3) \text{ are fulfilled and } f_l \in \mathbb{N} \text{ for all } l \in \mathscr{L} \right\}.$$

Minimizing the costs reflects an operators' perspective, whereas the following model is designed from passengers' view. A vehicle capacity A, bounding the number of passengers in a vehicle, is assumed to be given. In the direct travelers model (**Direct**) the aim is to maximize the number of passengers that travel directly on preferable paths, i.e., that do not have to transfer between lines:

$$(\textbf{Direct}) \qquad \begin{array}{ll} \{\max \text{ number of direct travelers} : & \text{the vehicle capacity} A \text{ is not exceeded,} \\ & (2) \text{ and } (3) \text{ are fulfilled,} \\ & \text{and } f_l \in \mathbb{N} \text{ for all } l \in \mathscr{L}\}. \end{array}$$

These two models, reflecting the conflicting perspectives outlined in the introduction, are used as reference models for line planning which allow the authors to evaluate the quality of a given line pool. The problem of finding a line pool is stated as follows.

Let $\mathscr{L}^0$ be the set of all cycle free paths on a given PTN $G = (V, E)$. Then the aim is to find a set $\mathscr{L} \subseteq \mathscr{L}^0$ such that the number of lines is bounded by given value $K \in \mathbb{N}$, i.e., $|\mathscr{L}| \leq K$, and constraints (2) and (3) are satisfied for all $e \in E$.

(**LPool**) Given the PTN $G = (V, E)$, lower and upper frequency bounds $f_e^{min} \leq f_e^{max}$ for all $e \in E$, and the set $\mathscr{L}^0$ of all cycle free paths in the PTN, find a set $\mathscr{L} \subseteq \mathscr{L}^0$ with $|\mathscr{L}| \leq K$ such that there exist $f_l \in \mathbb{N}$ for all $l \in \mathscr{L}$ with

$$f_e^{min} \leq \sum_{\substack{l \in \mathscr{L} \\ e \in l}} f_l \leq f_e^{min} \quad \text{for all } e \in E.$$

For (**LPool**) the authors show the complexity.

Theorem 2.1 *([GHS15b]) Problem (**LPool**) is NP-hard, even if $K = 1$.*

The proof reduces the hamiltonian path problem to (**LPool**). Despite the complexity, on a linear network this problem has an interesting property.

Proposition 2.2 *Let the PTN be a linear graph $G = (V, E)$, $f_e^{min} > 0 \ \forall e \in E$, $K = \infty$, $\mathscr{L}^0$ the set of all paths in G and the line-costs are of the form $cost_l = F + \sum_{e \in E} c_e$. Here F is a fixed cost-term and c_e are edge-dependent costs. Let l be the line consisting of all edges.*

1. *Then there exists an optimal solution to (**Cost**) which contains line l with frequency $f_l = \min_{e \in E} f_e^{\min}$*
2. *Then there exists an optimal solution to (**Direct**) which contains line l if the upper edge frequencies $f_e^{\max}$ are large enough.*

This proposition shows that the line containing all edges is contained in the line concept of (**Direct**) if the frequencies bounds are set appropriately. This property is used in the subsequently presented algorithm which is designed to solve the line pool generation problem. The algorithm is based on the idea of repeatedly generating minimal spanning trees of the infrastructure network and determining leaf and terminal nodes in that tree. A *leaf* node is a node of the network which is only adjacent to one edge. A *terminal* node is a node which is incident to more lines of the line pool (or edges if the line pool is empty) than a given threshold, specified by a parameter.

1. The algorithm starts with an empty line pool. Based on a minimal spanning tree of the PTN, w.r.t. the edge lengths, the algorithm determines leaf and terminal nodes. Then, lines are generated between either two leaves, two terminals, or a terminal and a leaf node, and are added to the line pool.

2. If, after that, conditions (2) and (3) cannot be satisfied with the current line pool, a different minimal spanning tree is generated. This minimal spanning tree preferably contains edges which prevent satisfying conditions (2) and (3). Then, the procedure starts over again, adding more lines to the line pool (go to 1.).

3. Otherwise, if conditions (2) and (3) can be satisfied by the current line pool or the maximal number of iterations is reached, the algorithm stops.

The authors analyze the outcome of the algorithm in a broad study on a network similar to the one of the German railways (Deutsche Bahn). For 624 different parameter settings line pools are generated and evaluated by executing the (**Cost**) model and the (**Direct**) model on them, provided within the LinTim framework. The quality of a line pool is judged by the costs, which is the objective of the (**Cost**) model, and the direct travelers, which is the objective of the (**Direct**) model. The following results are found.

For only 13% of all parameter settings the algorithm stops and does not return a feasible line pool. Additionally, particular parameter settings were identified which caused most of the infeasible pools. For all other parameter settings the algorithm returns line pools for which the subsequent (**Cost**) and (**Direct**) models could be solved in 2.20 and 11.97 seconds on average. Hence, the first result is that the algorithm reliably returns line pools which serve as input for subsequent line planning models. In more detail the following results were assessed.

- The more edges per line are enforced, the higher the number of direct travelers.
- Labeling fewer nodes as terminals increases the number of direct travelers and slightly reduces the costs.
- Enforcing a distance between the end nodes of a line increases the number of direct travelers.
- The size of the line pools did not have a big impact on their quality. Rather the choice of which lines to add to the line pool makes an impact on the line pool quality.

As the main objectives, by which the quality of a line pool is evaluated, are the costs and the number of direct travelers, this work can be classified as considering both company's perspective and passengers' interests.

3.3 Delay Resistant Line Planning with a View Towards Passenger Transfers

by J. Harbering

Revision submitted to the Journal of Transportation and Logistics, 21 *pages paper*

The basic planning problem solved in this work is the line planning problem, or more precisely the frequency setting problem (see Section 2.2). From the previous planning phases a PTN with lower and upper frequency bounds for all edges and a line pool are

given. In this problem frequencies for all lines from the line pool are sought such that conditions (2) and (3) are satisfied. The task is to plan the paths of passengers on top of the line planning problem. However, the aim is rooted in the final planning phase, the delay management problem. Thus, for motivating this line planning problem, the basic elements which make up the delay management problem are discussed first. In public transport, delays are spreading along passenger transfers and due to capacity constraints, e.g., on track segments. As outlined in the literature overview in Section 2.3, an attempt to combat the spreading is to plan buffer times in the timetable. Then, if delays occur, they can be buffered by the buffer times and thus do not spread as heavily. This approach of planning buffer times in the timetable increases passengers' traveling time. Thus, adding buffer to the timetable immediately leads to the trade-off between planned (without delays) and actual (with delays) traveling time (see [CT12]). In this work the author tries to avoid facing this trade-off but aims at combating possible delay spreadings even earlier. Since delays spread along passenger transfers, the idea is to minimize passenger transfers at the stage of line planning.

The formalization of the problem is given as follows. From the previous planning phases given, let $G = (V, E)$ be a public transportation network (PTN) which can be an output of the work summarized in Section 3.1. Furthermore, edge frequency constraints for all edges, i.e., (2) and (3), are given with the PTN. A set of possible lines $\mathscr{L}$ is assumed to be given, which can be obtained from the work summarized in Section 3.2. Furthermore, an $OD-$matrix C and a vehicle capacity A are given together with a set of preferable paths $\mathscr{P}_G^{ij}\ \forall i, j \in V$, i.e., for each passenger, on G. The set of preferable paths are paths in the PTN that can be chosen according to passengers' wishes. The author proposes to take the k_1-shortest paths with some $k_1 \in \mathbb{N}$ and $k_1 \geq 1$. In principle, any set of paths could be considered here. Note that the intention of the lower frequency bound constraint (2) is to ensure that all passengers can travel in the network. In the model presented here the passenger travel paths are contained in the model which ensures to provide sufficient capacity for the passengers. Hence, the lower frequency bound (2) can be dropped. Then the problem statement is given by the following.

(**LPT**) Given a PTN, $G = (V, E)$, a line pool $\mathscr{L}$, a set of preferable paths $\mathscr{P}_G^{ij}\ \forall i, j \in V$ on G, an $OD-$matrix C and the vehicle capacity A. Find a feasible line concept $(\mathscr{L}, f)$ such that (3) is respected and the number of passenger transfers is minimized.

Since a solution to (**LPT**) only gives a set of lines together with their frequencies, and not a timetable, the actual paths cannot be determined. Thus, only a concept for an

estimation of passenger paths can be developed at this stage. The author develops an estimation of the passenger transfers based on the concept of the Change&Go-Graph. For the construction of the Change&Go-Graph see [SS06].

Let $CG = (\mathscr{V}, \mathscr{D})$ be such a Change&Go-Graph, consisting of nodes $\mathscr{V}$ and edges $\mathscr{D}$. The set of edges $\mathscr{V}$ consist of OD-nodes $\mathscr{V}_{OD} \subseteq \mathscr{V}$ representing the start and end of passenger travels and line nodes $\mathscr{V}_{line} \subseteq \mathscr{V}$ representing the halting of trains at stations. The set of edges $\mathscr{D}$ is composed of OD-edges $\mathscr{D}_{OD} \subseteq \mathscr{D}$, line edges $\mathscr{D}_{line} \subseteq \mathscr{D}$, and transfer edges $\mathscr{D}_{transfer} \subseteq \mathscr{D}$. The OD-edges, line edges and transfer edges represent the boarding and alighting of passengers, the driving of trains and the transfers of passengers, respectively.

Let $p = (v_1, e_1, \ldots, v_{t_p-1}, e_{t_p-1}, v_{t_p}) \in \mathscr{P}_G^{ij}$ be a path on G with $v_1, \ldots, v_{t_p} \in V$ and $e_i = (v_i, v_{i+1}) \in E$ for all $i = 1, \ldots, t_p - 1$ and t_p being the number of nodes passed. Even more, we have $v_1 = i$ and $v_{t_p} = j$. This path does not specify which lines are used. Let a path $\mathfrak{p} = (\mathfrak{v}_1, \mathfrak{d}_1, \mathfrak{v}_2, \mathfrak{d}_2, \ldots, \mathfrak{v}_{r_\mathfrak{p}-1}, \mathfrak{d}_{r_\mathfrak{p}-1}, \mathfrak{v}_{r_\mathfrak{p}})$ on CG be such that $\mathfrak{v}_1, \ldots, \mathfrak{v}_{r_\mathfrak{p}} \in \mathscr{V}$, $\mathfrak{d}_i = (\mathfrak{v}_i, \mathfrak{v}_{i+1}) \in \mathscr{D}$ for all $i = 1, \ldots, r_\mathfrak{p} - 1$ and $r_\mathfrak{p}$ being the number of passed nodes. A path $\mathfrak{p}$ is a realization of p on CG if $\mathfrak{v}_1 = (i, 0), \mathfrak{v}_{r_p} = (j, 0)$ and $\{e \in E \mid (e, l) \in \mathfrak{p}\} = \{e_1, \ldots, e_{t_p}\}$.

Let $\mathscr{P}_{CG}^{ij}$ be the set of paths $\mathfrak{p}$, which are realizations of paths $p \in \mathscr{P}_G^{ij}$ on CG, and let $\mathscr{P}_{CG}$ be the union of those, i.e., $\mathscr{P}_{CG} = \bigcup_{i,j \in V} \mathscr{P}_{CG}^{ij}$. For a path $\mathfrak{p}$ on CG the number of transfers $c_\mathfrak{p}$ is given as the number of edges in $\mathfrak{p}$ which are elements of the set of transfer edges. Hence, $c_\mathfrak{p} = |\{d \in \mathfrak{p} \ : \ d \in \mathscr{D}_{transfer}\}|$.

Then, as an extension from [BKZ97] the IP formulation contains the following variables. Let $d_\mathfrak{p}$ be the number of passengers on path $\mathfrak{p} \in \mathscr{P}_{CG}$ and f_l be the frequency of line $l \in \mathscr{L}$.

The IP formulation is given as

$$(\textbf{IP-LPT}) \qquad \min \sum_{i,j \in V} \sum_{p \in \mathcal{P}_{CG}^{ij}} d_p c_p \tag{4}$$

$$\sum_{p \in \mathcal{P}_{CG}^{ij}} d_p \geq C_{ij} \quad \forall i,j \in V \tag{5}$$

$$\sum_{i,j \in V} \sum_{\substack{p \in \mathcal{P}_{CG}^{ij} \\ (e,l) \in p}} d_p \leq A f_l \quad \forall l \in \mathscr{L}, \forall e \in l \tag{6}$$

$$\sum_{\substack{l \in \mathscr{L} \\ e \in l}} f_l \leq f_e^{max} \quad \forall e \in E \tag{7}$$

$$d_p \in \mathbb{N}_0 \quad \forall p \in \mathcal{P}_{CG} \tag{8}$$

$$f_l \in \mathbb{N}_0 \quad \forall l \in \mathscr{L} \tag{9}$$

For a more elaborate derivation of the IP formulation see [Har15]. The complexity of (**LPT**) is discussed in the following theorem.

Theorem 3.1 *([Har15]) (**LPT**) is NP-complete, even if we have only one OD-pair.*

The proof is deduced as a reduction from the set partitioning problem. Additionally to the NP-hardness, the number of variables d may be of exponential size.

Lemma 3.2 *([Har15]) The number of variables may be exponential in the number of nodes $|V|$. This is even true if for all $i,j \in V$ the preferable path set $\mathcal{P}_G^{ij}$ only contains one element.*

Only if the number of transfers, which are allowed in a path, are bounded, the number of variables d is of polynomial size.

Lemma 3.3 *([Har15]) Suppose $|\mathcal{P}_G^{ij}| = 1$ for all $i \neq j \in V$ and a maximal number of transfers on each path $p \in \mathcal{P}_{CG}$ is fixed to s then the number of variables is of order $\mathcal{O}\left(|V|^{s+2}\right)$.*

The above theoretical results show that solving the IP program on any realistically sized instance is far too difficult. Hence, the author develops a promising approach based on column generation for the linear relaxation of (**IP-LPT**). A column generation procedure is also proposed in [BGP07] for a slightly different model. In Section 3.2 it is discussed that the line pool must not be too large in order to ensure computability. This means

that it can be assumed that the set of possible lines $\mathscr{L}$ is sufficiently small and hence only columns for the passenger path variables are generated.

The column generation decomposes the task of finding an optimal solution into the following steps.

1. Provide a feasible starting solution: A set of initial variables has to be computed which serves to generate a feasible starting solution. To this end, the set $\mathscr{P}_G^{ij}$ as the k_1-shortest paths ($k_1 \in \mathbb{N}$) in G (with respect to the edge-lengths) is computed for every OD-pair (i, j) with $i, j \in V$. Then, for each $i, j \in V$ and for each of path $p \in \mathscr{P}_G^{ij}$ the k_2-shortest paths ($k_2 \in \mathbb{N}$) in CG (with respect to the number of transfers) are computed which realize p on CG. The shortest path computations uses a variant of the Bellman-Ford algorithm [Bel56] which is implemented in [NC15]. This gives the set $\mathscr{P}_{CG}$.

2. Solve the master program: In this step the LP-relaxation of (IP-LPT) is solved by Xpress [Opt07].

3. Compute the pricing: In this step, promising paths in CG are repeatedly computed. The assessment of promising paths (paths that improve the current objective value) is a result from the dual of the solution to the master program. In the program studied in this work, the pricing comes down to computing the shortest path for all pairs in CG, for which the edge weights are obtained from the dual of the master program. If, in this step, the pricing returns promising paths, i.e., paths that improve the current objective value, they are added to $\mathscr{P}_{CG}$. For each new path a variable is added to the master program. Then go to 2. Otherwise an optimal solution is found and thus the column generation is stopped. Then go to 4.

4. Recover integer solution: The solution from the previous step is only optimal for the LP-relaxation of (IP-LPT). Since it is aimed for frequencies of lines, fractional numbers are not allowed. The approach which returns integer numbers takes the most promising path variables from the optimal LP-solution and solves the program with these as a MIP. The integer program is solved by Xpress. Note that the passenger variables are left fractional since these numbers are usually very high and only serve for planning purpose. Hence, these numbers do not have to be very precise.

Note that the described approach neither ensures feasibility nor optimality of the obtained solution. Still the results indicate a good performance with respect to both.

Numerical tests are conducted on different data sets and the performance of the column generation is evaluated within the LinTim framework. The following summarized results are obtained. On small instances the IP-formulation (**IP-LPT**) can be expressed and solved and here it returns the same objective value as the column generation approach. Hence, for those small instances the column generation proves to return the optimal solution. Additionally, in all experiments on larger instances the column generation returns a feasible solution. The number of passenger paths is not too big to be able to compute a solution and it is big enough to admit a feasible solution.

Finally, numerical experiments are conducted which study the aim for this work, i.e., to combat delay spreading. For different line planning models, such as (**Cost**), (**Direct**), and (**LPT**), line concepts are computed with LinTim. Then for each of these line concepts a timetable of comparable quality is computed. The different line concepts together with a timetable each compose different transportation systems. These systems receive equal delays and the delay resistance is accounted for. The model discussed in this approach is shown to return line concepts which belong to the most delay resistant ones. This final experiment concludes the work, stating that the adopted approach, minimizing passenger transfers at the stage of line planning, indeed leads to a delay resistant system. It shows that, when designing a system with delay resistance in mind, one should very much focus on the passenger transfers.

The approach presented in this publication has a main focus on minimizing passenger transfers. Additionally, the aim is to combat delay spreading. Both interests are mainly elements of the passenger convenience, as outlined in the introduction. However, delays have to some degree an influence on the efficiency of the railway system and thus also attributes to the companies' view. Finally, restricting the number of trains on an edge, here with the discussed frequency bounds, also ensures that costs are not arbitrarily high. Hence, this approach mainly focuses on passenger convenience elements but does neither lose sight of the costs.

3.4 Single Track Train Scheduling

by J. Harbering, A. Ranade, M. Schmidt, O. Sinnen
Published at MISTA 2015 Proceedings, 16 *pages paper*
Submitted to Journal of Scheduling, 20 *pages paper*

This work is inspired by railway problems arising in India. The Indian railway network and its structure is very much different from European networks and their operators'

performance.

In 2012, the total track length of the Indian railway system (64600 km) was twice as long compared to the German (33509 km) or the French (30013 km) network. The relative amount of single track stretches was much bigger on the Indian network than on the German or French network. In India about 70% of the network consisted of single track segments. In Germany and France the amount of single track stretches made up 56% and 43% of the network, respectively. When looking at the figures of ridership a problem of the Indian railways becomes evident. The Indian railways accounted for 1 trillion passenger kilometers, which is the number of passengers transported multiplied with the average traveling distance. In Germany and France these figures were considerably smaller with 80 and 85 billion, respectively. Data can be found in [TWB15, IUoR12, MoR13].

The figures indicate that a huge amount of passengers travel within the Indian network and the single track segments in the network pose bottlenecks for improving the performance. The aim of this work is to increase the usage of single track segments.

A literature overview of timetabling and related machine scheduling problems is given in Section 2.3. Note that event-activity networks, which are widely used to model timetabling problems, have also been used to model machine scheduling problems before (see [Seg74]).

This timetabling problem fits into the sequence of planning problems with the following considerations. It can be seen as an extension of the previously summarized works. From the previous planning phases a PTN, frequency bounds and a line concept, i.e., a set of lines with associated frequencies, are given. The result from the network design and the line planning phases can be assumed such that exactly the train lines of the problem discussed in this section are obtained. Assume the PTN given from the network design to be linear and OD information is specified from one end station to the other and vice versa. Then set the lower (and upper) frequency bound smaller (and bigger) than the number of trains needed. By solving the line pool generation problem (**LPool**) with a passenger focus and the subsequent line planning problem (**LPT**), only lines are obtained which traverse the entire linear network. The principle idea for the reasoning is given in Proposition 2.2 in the line pool generation work ([GHS15b]).

In the following the formal statement of the problem is given for which some notation is needed. Let a graph $G = (V, B)$ be given with V being the set of $n+1$ stations $v_1, \ldots, v_{n+1}$. Figure 3 depicts an example of a graph G. The stations are connected by the edges from B, which are n single track segments $b_{1,2} = (v_1, v_2), \ldots, b_{n,n+1} = (v_n, v_{n+1})$ (also called blocks). The traversal times $t_{i,i+1}$ on block $b_{i,i+1}$ for all $i = 1, \ldots, n$ are independent of the direction. By convention, let v_1 be the left-most station and, accordingly, v_{n+1} the

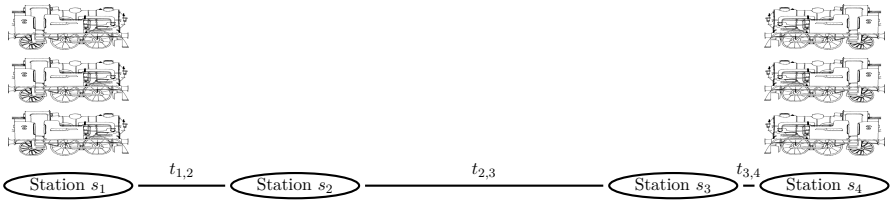


Figure 3: Example of graph underlying (STTS)

right-most station. Furthermore, let P^l be the number of trains from the left side (starting at v_1 and heading for v_{n+1}) P^r be the number of trains from the right side (starting at v_{n+1} and heading for v_1) is given. This can be viewed as having P^l right-bound and P^r left-bound directed lines. The constraints are the following. The block capacity is one, which means that a block is allowed to be traversed by at most one train at any point in time, i.e., there can be no two trains at the same time on the same block. No waiting times of trains at stations nor station capacities are considered. The aim is to minimize the makespan which is the time between the first departure of the first train and the last arrival of the last train.

The problem statement is given by the following.

(**STTS**) Let the graph $G = (V, B)$ with traveling times $t_{i,i+1}$ for all $i = 1, \ldots, n$, block capacities of one and no station capacities. Then the aim is to minimize the makespan for the traversal of G for P^l trains from left to right and P^r trains from right to left.

The purpose of this work is to study the complexity of (**STTS**) in general and under certain additional assumptions. For this aim, results from similar machine scheduling settings are discussed. Let therefore $t_{(i,i+1),j}$ be the traversal time of train $j \in \{1, \ldots, P^l + P^r\}$ on block $b_{i,i+1}$. In the classification scheme of scheduling problems (**STTS**) is described as a *job shop problem with two counter routes and no preemption* ($F^{\pm}n/t_{(i,i+1),j} = t_{i,i+1}/C_{max}$). The abbreviation stands for a flow shop problem (F) with two counter routes ($\pm$) on n blocks, traversal times independent of the train $j \in \{1, \ldots, P^l + P^r\}$ and the direction ($t_{(i,i+1),j} = t_{i,i+1}$) subject to minimizing the makespan (C_{max}). From the machine scheduling perspective, it means that the blocks are viewed as machines and the trains as jobs.

Under certain conditions, complexity results are known [Sev94]. For arbitrary traversal times and $n = 2$ the problem is solvable in polynomial time but for any $n \geq 3$, the problem is NP-hard [Jac56]. The latter result in particular holds if $P^l = P^r$ [GJS76].

In order to show complexity results, the authors develop a lower bound $T^{lo}(I)$ based on an instance I of the problem (**STTS**):

$$T^{lo}(I) = \max_{i=1,\dots,n} \left\{ \left(P^l + P^r\right) t_{i,i+1} + 2 \min \left\{ \sum_{j=1}^{i-1} t_{j,j+1}, \sum_{j=i+1}^{n} t_{j,j+1} \right\} \right\} \tag{10}$$

The block for which the maximum in $T^{lo}(I)$ is attained is called the *bottleneck* block. The correctness of the lower bound $T^{lo}(I)$ is shown in the next lemma.

Lemma 4.1 *([HRSS15]) $T^{lo}(I)$ is a lower bound on the objective value of an instance I of (**STTS**).*

Furthermore, an upper bound $T^{up}(I)$ based on an instance I of the problem (**STTS**) on the makespan is stated.

$$\begin{aligned} T^{up}(I) = \max_{i^* \in \{2,\dots,n-1\}} \Bigg\{ & \max \left\{ \sum_{i=1}^{i^*-1} t_{i,i+1} + \left(P^l - 1\right) \max_{i=1,\dots,i^*-1} t_{i,i+1}, \sum_{i=i^*+1}^{n} t_{i,i+1} + (P^r - 1) \max_{i=i^*+1,\dots,n} t_{i,i+1} \right\} \\ & + \left(P^l + P^r\right) t_{i^*,i^*+1} \\ & + \max \left\{ \sum_{i=1}^{i^*-1} t_{i,i+1} + (P^r - 1) \max_{i=1,\dots,i^*-1} t_{i,i+1}, \sum_{i=i^*+1}^{n} t_{i,i+1} + \left(P^l - 1\right) \max_{i=i^*+1,\dots,n} t_{i,i+1} \right\} \Bigg\} \end{aligned} \tag{11}$$

Again, its correctness is shown.

Lemma 4.2 *([HRSS15]) $T^{up}(I)$ is an upper bound on the objective value of an instance I of (**STTS**).*

If, for a given problem of type (**STTS**), it can be shown that a particular strategy attains $T^{lo}(I)$ as objective value, the optimality of that strategy is proved.

The following results state versions of the problem (**STTS**) for which a strategy is given which attains the objective value $T^{lo}(I)$. The first results assumes all traversal times to be equal, i.e., $t_{i,i+1} = 1$ for all $i = 1, \dots, n$. In the machine scheduling context this property is called *unit processing times*.

Theorem 4.3 *([HRSS15]) For instances of (**STTS**) with unit processing times, the makespan of an optimal schedule is given by $T^{lo}(I)$. This even holds if the capacity at all stations is bounded by three.*

The following lemma restricts the capacity of a station, i.e., the number of trains which are allowed to be at a station at any point in time, to two.

Lemma 4.4 *([HRSS15]) If the station capacity is restricted to two, the makespan of an optimal schedule is given by $T^{lo}(I)$ if there are unit processing times and $P^l = P^r$.*

The optimal strategy essentially schedules all trains with a headway of two time units from both ends. This is shown to result in a makespan of $T^{lo}(I)$, i.e., it is optimal. Finally, restricting the station capacity to one gives the following result.

Lemma 4.5 *([HRSS15]) For instances of (**STTS**) with unit processing times, and the station capacity restricted to one, the makespan of an optimal schedule is given by $2n - 1 + P^l + P^r$.*

The restriction to the capacity of one means that no two trains can pass each other. Hence, scheduling all trains from one side first and subsequently from the other side gives the optimal strategy. The next results state the complexity of (**STTS**) when bounding the number of blocks to three. For the following considerations unit processing times are no longer assumed.

Theorem 4.6 *([HRSS15]) For instances I of (**STTS**) with three blocks and $P^l = P^r$, the makespan of an optimal schedule is given by $T^{lo}(I)$.*

The proof provides a schedule for the case that the lower bound $T^{lo}(I)$ is achieved on the central bottleneck block and then uses this result for the other cases. For the instance in Theorem 4.6, dropping the requirement of equal number of left- and right-bound trains, the authors show the following weakly polynomial complexity.

Theorem 4.7 *([HRSS15]) Instances I of (**STTS**) with three blocks, i.e., $n = 3$, can be solved in*

$$\mathcal{O}\left(\left(P^l + P^r\right)^2 \left(\log^2\left(P^l + P^r\right) + \log\left(P^l + P^r\right) \log\left(\max\{t_{1,2}, t_{3,4}\}\right)\right)\right) \tag{12}$$

The strategy in the proof fixes the schedule of all trains on the outer blocks and thus reduces the problem to a single machine scheduling problem on the central block. On this block it is a scheduling problem with release and due dates which can be efficiently solved in polynomial time by the algorithm presented in [Sim78].

Finally, the authors show a pseudo-polynomial algorithm for the general problem (**STTS**). The proof uses a dynamic programming formulation which is expressed by an acyclic directed graph. First the construction of the graph is shown and subsequently the number of nodes and arcs is analyzed.

Lemma 4.8 *([HRSS15]) If block lengths are integer, the acyclic directed graph which represents the dynamic programming formulation has*

$$O\left(\binom{P^l+n}{n}\binom{P^r+n}{n}\max\{t_{i,i+1}\}\right) \quad \textit{nodes and}$$
$$O\left(3^n\binom{P^l+n}{n}\binom{P^r+n}{n}\max\{t_{i,i+1}\}\right) \quad \textit{arcs.}$$

The application of a shortest path algorithm to the acyclic directed graph from Lemma 4.8 gives the following complexity (see [Cor09]).

Lemma 4.9 *(**STTS**) can be solved in* $O\left(3^n\binom{P^l+n}{n}\binom{P^r+n}{n}\max\{t_{i,i+1}\}\right)$.

Finally, the authors obtain the pseudo-polynomial complexity result.

Corollary 4.10 *([HRSS15]) For a fixed number of blocks, the (**STTS**) can be solved in polynomial time in* P^l, P^r, *and* $\max_i t_{i,i+1}$.

Note that the algorithm in Theorem 4.7 has significantly lower runtime than the one in Lemma 4.8. The advantage of the dynamic programming formulation is that various extensions can be included into the graph construction adapting this to more realistic conditions. For example, enforcing minimal waiting times of trains at stations or station capacities can be handled within the dynamic programming formulation without changing the complexity result.

The aim in this work is to improve the performance of the operator, by having as many trains as possible traveling within a given amount of time. However, this can also be seen (similar to the other summarized works) as another passenger oriented perspective. As indicated by the statistical figures from India, the trains are heavily loaded. Increasing the usage of single track segments allows more trains and thus also more passengers to travel on the network. Additionally, the accommodated passengers can travel more comfortably by spreading among the available trains and thus decreasing the crowding.

3.5 Network-based Source Detection for Propagation Processes on Complex Networks with Application to Train Delays on Railway Systems

by J. Manitz, J. Harbering, M. Schmidt, T. Kneib, A. Schöbel
Published at IWSM 2014 Proceedings, 5 *pages paper*
Submitted to the Journal of the Royal Statistical Society: Series C, 48 *pages paper*

From the previous planning phases a planned timetable for trains on an infrastructure network is given. However, as outlined in the literature overview in Section 2.4, timetables can usually not be operated exactly as planned due to unforeseen exterior disturbances and disruptions. These disturbances can be weather related, due to infrastructure failure, late staff, or many other reasons. Such disturbances introduce delays into the timetable which the operator has to deal with. For the propagation it holds that trains cannot drive faster than allowed and a certain minimal time has to be ensured for transferring passengers. Thus, the constraints for the disposition timetable (timetable under delays) are the lower bounds on the duration of the activities for which the timetable is planned. The operator not only has to decide upon the disposition times. He also has to decide which of the planned transfers are to be maintained (*wait-depart* decisions) and how to sequence the trains on the capacitated block sections (*priority* decisions). The aim is to minimize the delay of the passengers at their final destination. A general rule stating how to take wait-depart and priority decisions is called a delay management rule. Other propagation mechanisms, such as crew- or vehicle-dependencies are neglected in this study.

Given a timetable and a station inflicting major delays to the timetable, a delay spreading process on the network results. The aim of this work is to determine the source of the spreading process, i.e., the particular station inflicting the delays, with the additional assumption that the timetable is not known. The delays which are to be detected in this work are called the *signal*. The determination of the source gives practitioners the ability to further investigate the reasons of why source delays were generated in that particular station. However, from a practical point of view, it is a difficult task to determine the source of a spreading process without the information of the timetable. This task is even more hindered by the fact that usually not only one station introduces delays into the system but also smaller delays are arising within the network in other stations. These

other delays can be understood as background noise delays, while the delays introduced from the main source station are the signal which is to be detected. Additionally, there might be more than just one station introducing signal delays. Hence, also a multiple source detection approach has to be facilitated.

The method for the detection of the source, used in this work, is adapted from [MKS+14] where it was applied to the EHEC/HUS disease outbreak in Germany in 2011. In this application the approach proved to return very reliable results detecting the source of the disease outbreak within the actual area only shortly after the first evidences of the disease outbreak. Assuming the spreading to be of a circular-like shape when redefining the network distance, the source is computed as the station giving the most circular shape.

More formally, given a network $G = (\mathscr{K}, \mathscr{L})$, with nodes $\mathscr{K}$ and edges $\mathscr{L}$, a stochastic process underlying the spreading of delays is assumed. This stochastic process is described as $X_k(t)$, which gives the relative delay of a train in a station $k \in \mathscr{K}$ until time $t \in [0, T]$. The time T is the time until which the spreading phenomenon is analyzed. The detected delays $x_k(t)$ are the realizations of the stochastic process $X_k(t)$. In this work the realizations were recorded at predefined points in time $t_1, \ldots, t_n \in [0, T]$ giving n sequential pictures of the spreading process.

Essential to this method is that the distances in the network are redefined. Instead of taking the Euclidean or geodesic distance the effective network distance is used. The effective network distance, introduced by [BH13], combines a deterministic and a probabilistic element into giving the distance of a path in the network G. Let Γ_{kl} be the set of all paths between $k, l \in \mathscr{K}$ and $L(\gamma_{kl})$ be the number edges in path $\gamma_{kl} \in \Gamma_{kl}$. Furthermore, let p_{kl} be the transition probability of going from node l to k. The distance $d_{\mathtt{eff}}(k, l)$ between two nodes $k, l \in \mathscr{K}$ is then given as the length of the shortest path between the two nodes computed with Dijkstra's algorithm [Dij59]:

$$d_{\mathtt{eff}}(k,l) \quad = \quad \min_{\gamma_{kl} \in \Gamma_{kl}} \left[L(\gamma_{kl}) - \sum_{(k_i, k_{i-1}) \in \mathscr{L}_{\gamma_{kl}}} \log p_{k_i k_{i-1}} \right]. \tag{13}$$

The length of an edge $(h, k) \in \mathscr{L}$ is thereby measured as $1 - log(p_{kl})$. The 1 represents the deterministic part, accounting for one edge, whereas p_{kl} can encode different effects. The transition probability p_{kl} for $k, l \in \mathscr{K}$ can encode edge weights according to passenger counts between l and k, the number of train rides between l and k, or it can be the number of edges incident to l. This definition of effective distance has proved to properly determine sources of spreading processes, see [MKS+14]. In this study, it is examined whether dif-

ferent information encoded in this probability has an effect on the detection performance. Given the realizations $x_k(t)$ for all $k \in \mathcal{K}$, the source k_0 of the spreading process at time t is determined as the station minimizing the mean of the weighted realizations, i.e.,

$$k_0(t) \in \arg\min_{k \in \mathcal{K}} \frac{1}{\sum_{l \in \mathcal{K}} x_l(t)} \sum_{l \in \mathcal{K}} x_l \, d_{\mathtt{eff}}(k, l). \tag{14}$$

This method is referred to as *network-based* source detection. Another method is developed within this study which specifically applies to the detection of delays in public transportation systems. It is based on the idea that delays mainly spread along a line which the affected train passes. Given a picture of the delay spreading process, delays are traced back from a station by checking whether the relative delay of a train is higher in a nearby station. If so, the algorithm proceeds with this station by checking its nearby stations. This method is referred to as *backtracking*. The last method the results are compared with, is referred to as *Comin's* method and is adapted from [CdFC11]. This method considers the subgraph of the network G, induced by the nodes of $\mathcal{K}$ which have experienced delays. From this subgraph, the node with the highest centrality measure, in terms of betweenness (see [Fre77]), is deduced as the source of the spreading process. For this we require that at least four stations have experienced delays.

With the help of the software tool LinTim the authors are able to generate a broad set of test instances to compare the performance of the different methods. In the standard setting, the test instance is the railway system of the major German railway operator, *Deutsche Bahn*. On this instance a line concept and a timetable are computed. Given the timetable, the authors assume one station to be the source of several source delays. These delays are introduced into the timetable and propagated with the following rule:

Trains wait for delayed transferring passengers not more than a given fixed amount of time and the sequence of trains on the blocks is left according to the planned timetable.

During the spreading of the delays within the subsequent four hours, ten sequential pictures are taken. In each such picture the relative amount of delay of a train in a station is computed. This relative amount is considered to be the realization of the stochastic process in that station. In each picture the source detection methods are applied to these realizations. The above described ten sequential pictures are generated for spreadings originating from every station, in turn, giving a set of $|\mathcal{K}|$ simulations.

In order to prove the method's applicability for practical purposes, the German railway operator *Deutsche Bahn* also provided real data instances on which the methods are tested. This data comprises delay information from three different days where at partic-

ular stations major delays are introduced into the system. Since this is real-world data, the methods also have to deal with noise from other delays.

The performances of the methods is accounted for by different measures. The most important measure is the *probability of correct detection* which counts the amount of correctly found sources at a given time instance. Next, for one picture all stations are ranked according to (14). In this list, the rank of the correct source is called the *rank of correct detection*. Note that this value is averaged among all simulations. The last performance measure determines the *distance to correct source*, i.e., the mean distance between the detected and the correct source. By distance it is here referred to the network distance with edge weights according to their Euclidean length.

Examining the performance of the methods in the standard setting, the authors show that the network-based method and the backtracking method are by far superior to Comin's method. Even more, the backtracking method seems to have slight advantages over the network-based method under laboratory conditions generated by LinTim where no background noise is present. Based on the standard setting, additional effects are examined in order to show the performances of the different methods. The authors recognize that the network-based method does not show large differences whether delays are propagating due to wait-depart decisions only or due to additional sequencing decisions. Also different delay management rules are tested. It is indicated that the differences among the performances of the network-based method under propagating delays with different delay management rules are small. Analyzing the detection for different delay management rules further, a strong correlation between the detection performance and the amount of data is recognized: the smaller the mean delay magnitude of the detected delays, the higher the probability of correct detection.

Additionally to the railway network of Germany, the network-based method was tested on the metro network of Athens and the bus network of Göttingen. These tests indicate that the performance improves for larger networks. The final test set comprises data in the standard setting extended by random noise. The authors show that the amount of noise has a significant impact on the detection performance, still showing that even though small amounts of noise exist, the source of the spreading can be detected reliably.

In addition to the laboratory conditions generated by LinTim, further insight on the methods performances is obtained by applying them to the real data provided by *Deutsche Bahn*. On three test instances from three different days with major disruptions in a single station, the source is detected. On this data the network-based method and the backtracking complement each other. During the first few time steps, the backtracking

achieves a good performance which decreases during the later time steps. Complement to this, the performance of the network-based method has a weaker performance during the first time steps and achieves good results during the later time steps. On some instances the number of stations which have experienced delays is less than four which means that Comin's method cannot be applied. On the other instances Comin's method shows weaker performance than the other methods.

The following conclusions are summarized from the test results. The backtracking method and the network-based method show similar performances with slight advantages to backtracking on test instances generated by LinTim. The network-based method shows robustness, i.e., similar strong performances, under various effects, such as the different consideration of delay spreading mechanisms and delay management rules. It is shown that the network structure and the amount of noise have a significant impact on the performance quality. Still, under large amounts of noise in the real-world data the network-based method and the backtracking method complement each other and show strong performances.

The result of this work, the network-based and the backtracking method, are not increasing the effectiveness of a railway system or improving passenger convenience. The idea is to develop a statistical method and thereby provide more insight into spreading patterns provoked by delays. If the propagation of delays can then be better understood, their spreading can be detected earlier and hence also prevented. In the long term, both costs decrease and passenger convenience increases. Hence, both the company and the passengers may benefit.

4. Discussion of Results

The overall contribution of this thesis is twofold. First, this thesis shows how the consecutive planning stages of public transportation are linked together by providing an overview of all problems and discussing a work for each of the main problems arising in this sequence. In particular, beginning with the network design problem in Section 2.1 and the additional generation of line pools (Section 3.2) the solution can be passed to the line planning problem in Section 3.3. Solving the line planning problem, a line concept is obtained which is the input for the timetabling problem. In the studied timetabling problem in Section 3.4, the line concept is simple. Still, one could assume that the line planning model in Section 3.3, based on a linear graph, leads to this concept. Then, delays are introduced into the operated timetable and propagated within the system allowing to study the arising propagation process in Section 3.5. This shows that an entire sequence of planning problems is represented by the works summarized in Section 3.

The second overall contribution is the passenger convenience viewpoint in all of the contributed works. In the first work in Section 3.1 the passengers' traveling time estimate is minimized while ensuring a certain maximal access and egress distance. In the line pool generation problem in Section 3.2, line pools are built with the focus of providing passenger friendly lines by aiming for many direct travelers. In the line planning problem in Section 3.3 the objective is to minimize passenger transfers and the aim is to study the implication on delay spreading. In this work it is shown that line concepts which allow that passengers have to transfer only few times grant more delay resistance which is very desirable from passengers' point of view. The timetabling problem in Section 3.4 then aims at providing as many services as possible in order to increase availability and decrease crowding. Finally, the last problem in Section 3.5, concerned with the delay spreading pattern, provides deeper insight into the way delays spread which facilitates combating the spreading more effectively in the long run. Hence, it can be summarized that different passenger aspects are represented in all of the studied works.

The single works have also made particular contributions. The novelty of the work in

the network design phase in Section 3.1 is the inclusion of the realistic passenger traveling time estimate. This is composed of a nonlinear speed function which takes into account the acceleration and deceleration, thus approximating the vehicle speed very precisely. It is shown that using this approach significantly different stop location patterns are obtained.

The connecting problem to the line planning phase, the line pool generation problem, summarized in Section 3.2, has been rarely studied before. Of those few works considering this problem, most authors use a simple shortest path approach. In the studied approach an algorithm is developed which uses minimal spanning trees of the public transportation network to compute appropriate lines. Different parameter influences are studied and their effect is analyzed with respect to passenger convenience and companies' point of view. Such an approach has not been studied before.

The line planning problem in Section 3.3 states a problem which has been investigated similarly before. The first difference to previous approaches is the extension of the preferable paths. In the (**Direct**) model for each passenger a preferable path is considered which is extended to include a set of preferable paths for each passenger. To the passengers, this allows greater freedom for the choice of the path. Other similar approaches ([BGP07, SS06]) intend to minimize the transfers by allowing any path for the passengers. Due to the capacity constraints this might result in bad paths for some of the passengers. Restricting the length of the paths would significantly increase the complexity. With the presented approach of a set of preferable paths this increase of complexity is avoided. The proposed solution approach is also applied to the model presented in [BGP07]. The additional value of the approach in Section 3.3 is the bigger instance size which can be solved.

Furthermore, in this work the effect on delay resistance is analyzed by computing a timetable on basis of the line concept and by introducing delays into the timetable. An analysis of the delay resistance of different line concepts has been conducted in [GSS13]. In [GSS13] the model (**LPT**) has not been considered. Hence, as an extension, in [Har15], it is shown that passenger-friendly line planning approaches, in terms of few transfers, and in particular the model (**LPT**), tend to provide a basis for more delay resistant transportation systems.

As outlined in the general literature overview in Section 2, there exists an increasing necessity and trend to study integrated approaches. With the aim of this model, i.e., providing delay resistance by incorporating passenger transfers in the objective function at the stage of line planning, two different planning problems are connected. While integration has the idea to connect problems even closer, this model connects the delay management

and the line planning problem.

The timetabling problem in Section 3.4 provides an analysis on how to use the available infrastructure optimally. A similar problem has been studied within the machine scheduling context and some complexity results are known for that problem. With the presented work the gap between settings under which the problem is NP-hard and settings under which it is solvable in polynomial time has been narrowed. It is therefore a contribution to understanding the fundamentals of single track train scheduling.

The last problem, analyzing delay spreadings, contributes to the understanding of the resulting propagation patterns. A method adopted from disease spreadings is shown to properly deduce the source of the spreading pattern, even on real data. Additionally, a new method which traces back the propagated delays is proposed. This method shows to be very applicable in the delay propagation case.

In the following section the final conclusions are stated and further research lines are highlighted.

5. Conclusion and Future Work

A contribution of this thesis is the passenger convenience perspective being represented in distinct ways in all of the works. Furthermore, the linkage between the problems composing the sequence of planning stages in public transportation is strong. It is shown how a problem builds on top of the previous problem and thus allows to plan an entire public transportation system.

The integration of the planning problems has been outlined in the general literature overview in Section 2. While only one of the presented works, namely the line planning problem, is an attempt to connect two different problems, there are still broad areas open for further research. Solving two subsequent planning problems by an integrated approach should thereby facilitate an even more thorough understanding of the dependencies and bottlenecks of the problems. For example, the network design problem could be combined with the line planning problem, i.e. designing the infrastructure network at the same time as finding promising lines and setting their frequencies. Another idea for extending the network design model, when considering passenger convenience, is to explicitly model passenger travel paths and plan the stops accordingly. An approach, given in [KMP+14], shows that locating two stops in dependence of passenger paths is already a difficult task. A further development of such an approach would provide the basis for a more accurate estimate of passengers' traveling time.

A further research line for the line planning model could be to generalize the objective function to any weighted traveling time function. As outlined in the introduction, one is mainly interested in minimizing passenger inconvenience, which is composed of various elements of which the transfers are just one. Hence, generalizing this beyond the traveling time line planning model ([SS06]) would allow to account for the passenger inconvenience. The column generation applicability for such a general scheme has to be shown.

The timetabling problem can still be studied further, in particular its complexity. In the discussed publication only the pseudo-polynomial complexity for the general case and the weakly polynomial time algorithm for the three block case are given. For this case no

proof exists, stating whether the weakly polynomial algorithm is optimal or whether there exists a strongly polynomial time algorithm. Further, it is still open whether (**STTS**) for an arbitrary number of blocks is NP-hard or not.

Finally, the methodology of the last work can be extended. As outlined, it is uncommon that only a single source exists which introduces major disruptions. Usually, multiple sources exist and the source detection methods have to be further developed in order to apply to these cases. A first approach decomposes the network by clustering stations into separate clusters and applies the single source detection method to each of these clusters. Furthermore, in order to assess the certainty that the detected source is the correct source, a measurement could be developed. A first attempt uses a subsampling strategy which selects only subsets of the entire data set and applies the source detection method on this narrow data set. Comparing the result for many different subsets with the result on the entire data set such a certainty assessment can be stated.

Another general further research line is to combine the planning problems with simulations. The power of simulations rises from the fact that such models can be engineered far more realistically. Still, in most cases it is not known what the reality is like and optimal solutions are very hard to obtain by simulations. Since the downside of optimization models is that, in order to preserve computability, most problems can not be modeled realistically, these two approaches could be combined. The idea is that the optimization module provides provable good solutions which are verified by the simulation to be feasible in a realistic case as well.

This thesis contributes to the linkage between the different planning stages in public transportation, provides an insight in new passenger convenience elements throughout the planning and points out further lines of research. There are still many interesting facets of mathematical problems to be studied in the area of public transportation planning in order to improve towards current and future passenger demands.

"An advanced city is not a place where the poor move about in cars, rather it's where even the rich use public transportation."

– Enrique Peñalosa,
Mayor of Bogotá,
President of the US Institute for Transportation and Development Policy (ITDP)

6. Own Contribution

In this section the own contribution by J. Harbering to the presented publications are summarized.

[CHS14] is a joint work with E. Carrizosa and A. Schöbel. Most of the problem analysis, including the proofs, and the programming was done by J. Harbering. His contribution can be judged to 50%.

[GHS15b] is a joint work with P. Gattermann and A. Schöbel. Note that [GHS15a] has received the Young Researcher Award on the Conference on Computer Aided Systems in Public transportation (CASPT). The laudatio particularly mentioned the writing style and the novelty of the topic as reasons for giving this award. The idea for this topic, the analysis of the results and its writing have been done by J. Harbering. His own contribution can be judged to 30%.

J. Harbering is the sole author of [Har15]. Hence, the idea for this work, the theoretical results and the experiments are entirely developed and conducted by J. Harbering.

[HRS15] is a joint work with M. Schmidt, A. Ranade, and O. Sinnen. Large parts of the paper, including the relation to machine scheduling, proofs and writing are contributed by J. Harbering. His own contribution can be judged to 50%.

[MHS+15] is a joint work with J. Manitz, T. Kneib, M. Schmidt, and A. Schöbel. The own contribution by J. Harbering consists in the idea of possible effects to study the methods on, generating and providing the data for the simulation study, the idea for the backtracking method and writing large parts of the manuscript. Thus, his contribution can be judged to 40%.

Bibliography

[Ban08] D. Banister. The sustainable mobility paradigm. *Transport policy*, 15(2):73–80, 2008.

[BBB77] A.I. Babushkin, A.A Bashta, and I.S. Belov. Scheduling for problems of counterroutes. *Cybernetics and Systems Analysis*, 13(4):611–617, 1977.

[BBK13] M. Bender, S. Büttner, and S.O. Krumke. Online delay management on a single train line: beyond competitive analysis. *Public Transport*, 5(3):243–266, 2013.

[Bel56] R. Bellman. On a routing problem. Technical report, DTIC Document, 1956.

[BGP07] R. Borndörfer, M. Grötschel, and M.E. Pfetsch. A column-generation approach to line planning in public transport. *Transportation Science*, 41(1):123–132, 2007.

[BGP08] R. Borndörfer, M. Grötschel, and M.E. Pfetsch. Models for line planning in public transport. In *Computer-aided systems in public transport*, pages 363–378. Springer, 2008.

[BH13] D. Brockmann and D. Helbing. The hidden geometry of complex, network-driven contagion phenomena. *Science*, 342(6164):1337–1342, 2013.

[BHLS10] A. Berger, R. Hoffmann, U. Lorenz, and S. Stiller. Online railway delay management: Hardness, simulation and computation*. *Simulation*, 2010.

[BKZ97] M.R. Bussieck, P. Kreuzer, and U.T. Zimmermann. Optimal lines for railway systems. *European Journal of Operational Research*, 96(1):54–63, 1997.

[BWZ97] M. R. Bussieck, T. Winter, and U.T. Zimmermann. Discrete optimization in public rail transport. *Mathematical programming*, 79(1-3):415–444, 1997.

[CCT10] V. Cacchiani, A. Caprara, and P. Toth. Non-cyclic train timetabling and comparability graphs. *Operations Research Letters*, 38(3):179–184, 2010.

[CdFC11] C.H. Comin and L. da Fontoura Costa. Identifying the starting point of a spreading process in complex networks. *Physical Review E*, 84(5), 2011.

[CFT02] A. Caprara, M. Fischetti, and P. Toth. Modeling and solving the train timetabling problem. *Operations research*, 50(5):851–861, 2002.

[CHK+14] V. Cacchiani, D. Huisman, M. Kidd, L. Kroon, P. Toth, L. Veelenturf, and J. Wagenaar. An overview of recovery models and algorithms for real-time railway rescheduling. *Transportation Research Part B: Methodological*, 63:15–37, 2014.

[CHS13] E. Carrizosa, J. Harbering, and A. Schöbel. The Stop Location Problem with Realistic Traveling Time. In Daniele Frigioni and Sebastian Stiller, editors, *13th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems*, volume 33 of *OpenAccess Series in Informatics (OASIcs)*, pages 80–93, Dagstuhl, Germany, 2013. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[CHS14] E. Carrizosa, J. Harbering, and A. Schöbel. The Stop Location Problem with Realistic Traveling Time. *Submitted to Journal of the Operational Research Society (JORS)*, 2014.

[Cor09] T.H. Cormen. *Introduction to algorithms*. MIT press, 2009.

[CPW99] B. Chen, C.N. Potts, and G.J. Woeginger. A review of machine scheduling: Complexity, algorithms and approximability. In *Handbook of combinatorial optimization*, pages 1493–1641. Springer, 1999.

[CS08] C. Conte and A. Schöbel. *Identifying dependencies among delays*. PhD thesis, PhD thesis, Niedersächsische Staats-und Universitätsbibliothek Göttingen, Germany, 2008.

[CT12] V. Cacchiani and P. Toth. Nominal and robust train timetabling problems. *European Journal of Operational Research*, 219(3):727–737, 2012.

[CTV98] J.-F. Cordeau, P. Toth, and D. Vigo. A survey of optimization models for train routing and scheduling. *Transportation science*, 32(4):380–404, 1998.

[CvDZ98] M.T. Claessens, N.M. van Dijk, and P.J. Zwaneveld. Cost optimal allocation of rail passenger lines. *European Journal of Operational Research*, 110(3):474–489, 1998.

[CW86] A. Ceder and N.H.M. Wilson. Bus network design. *Transportation Research Part B: Methodological*, 20(4):331–344, 1986.

[dBBB+13] J. Van den Bergh, J. Beliën, P. De Bruecker, E. Demeulemeester, and L. De Boeck. Personnel scheduling: A literature review. *European Journal of Operational Research*, 226(3):367–385, 2013.

[DDW09] Z. Drezner, T. Drezner, and G.O. Wesolowsky. Location with acceleration–deceleration distance. *European Journal of Operational Research*, 198(1):157–164, 2009.

[DH07] G. Desaulniers and M.D. Hickman. Public transit. *Handbooks in operations research and management science*, 14:69–127, 2007.

[Dij59] E.W. Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.

[Dus88] B.I. Dushin. An algorithm for the solution of the two-route johnson problem. *Cybernetics and Systems Analysis*, 24(3):336–343, 1988.

[Eur15] Eurostat. Main tables. website, 2015. Available online: http://ec.europa.eu/eurostat/web/transport/data/main-tables. Last access 28-10-2015.

[FM10] L. Fan and C.L. Mumford. A metaheuristic approach to the urban transit routing problem. *Journal of Heuristics*, 16(3):353–372, 2010.

[Fre77] L.C. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, pages 35–41, 1977.

[GHS15a] P. Gattermann, J. Harbering, and A. Schöbel. Generation of Line Pools. In *Proceedings of the 13th Conference on Advanced Systems in Public Transport CASPT*, 2015.

[GHS15b] P. Gattermann, J. Harbering, and A. Schöbel. Generation of Line Pools. *Public Transport*, 2015. submitted.

[GHS15c] M. Goerigk, J. Harbering, and A. Schöbel. LinTim - Integrated Optimization in Public Transportation. Homepage. see http://lintim.math.uni-goettingen.de/, 2015.

[GJS76] M. R. Garey, D. S. Johnson, and R. Sethi. The Complexity of Flowshop and Jobshop Scheduling. *Mathematics of Operations Research*, 1976.

[GMMW09] M. Gatto, J. Maue, M. Mihalák, and P. Widmayer. Shunting for dummies: An introductory algorithmic survey. In *Robust and Online Large-Scale Optimization*, pages 310–337. Springer, 2009.

[Gov98] R.M.P. Goverde. Max-plus algebra approach to railway timetable design. In *The 1998 6 th International Conference on Computer Aided Design, Manufacture and Operation in the Railway and Other Advanced Mass Transit Systems*, pages 339–350, 1998.

[GSS13] M. Goerigk, M. Schachtebeck, and A. Schöbel. Evaluating line concepts using travel times and robustness. *Public Transport*, 5(3):267–284, 2013.

[Har15] J. Harbering. Delay Resistant Line Planning. *Submitted to EURO Journal on Transportation and Logistics (EJTL)*, 2015.

[HDD01] K. Haase, G. Desaulniers, and J. Desrosiers. Simultaneous vehicle and crew scheduling in urban mass transit systems. *Transportation Science*, 35(3):286–303, 2001.

[HKLV05] D. Huisman, L.G. Kroon, R.M. Lentink, and M.J.C.M Vromans. Operations research in passenger railway transportation. *Statistica Neerlandica*, 59(4):467–497, 2005.

[HLS+01] H.W. Hamacher, A. Liebers, A. Schöbel, D. Wagner, and F. Wagner. Locating new stops in a railway network. *Electronic Notes in Theoretical Computer Science*, 50(1), 2001.

[HRS15] J. Harbering, A. Ranade, and M. Schmidt. Single Track Train Scheduling. In *Proceedings of the 7th Multidisciplinary International Scheduling Conference: Theory and Applications MISTA*, 2015.

[HRSS15] J. Harbering, A. Ranade, M. Schmidt, and O. Sinnen. Single Track Train Scheduling. *Journal of Scheduling (JOS)*, 2015. Submitted.

[Hui04] D. Huisman. *Integrated and dynamic vehicle and crew scheduling.* Erasmus School of Economics (ESE), 2004.

[ITF15] International Transport Forum. About. website, 2015. Available online: http://www.internationaltransportforum.org/. Last access 27-10-2015.

[IUoR12] International Union of Railways. Synopsis 2012, 2012. Available online: http://old.uic.org/spip.php?article1347. Last access 20-10-2015.

[Jac56] J.R. Jackson. An extension of johnson's result on job lot scheduling. *Naval Res. Logist.*, 1956.

[JGPC+09] J. Jespersen-Groth, D. Potthoff, J. Clausen, D. Huisman, L. Kroon, G. Maróti, and M.N. Nielsen. *Disruption management in passenger railway transportation.* Springer, 2009.

[KK09] K. Kepaptsoglou and M. Karlaftis. Transit route network design problem: review. *Journal of transportation engineering*, 2009.

[KK15] F. Kirchhoff and M. Kolonko. Modelling delay propagation in railway networks using closed family of distributions. *Technical Report*, 2015.

[KMP+14] M.-C. Körner, J.A. Mesa, F. Perea, A. Schöbel, and D. Scholz. A maximum trip covering location problem with an alternative mode of transportation on tree networks and segments. *Top*, 22(1):227–253, 2014.

[KWH75] P. Kolesar, W. Walker, and J. Hausner. Determining the relation between fire engine travel times and travel distances in new york city. *Operations Research*, 23(4):614–627, 1975.

[Lie07] C. Liebchen. *Periodic Timetable Optimization in Public Transport.* Springer, 2007.

[Lie08] C. Liebchen. Linien-, fahrplan-, umlauf-und dienstplanoptimierung: wieweit koennen diese bereits integriert werden? In *In HeurekaâĂŹ08 âĂŞ Optimierung in Transport und Verkehr, Tagungsbericht.*, 2008.

[LJMO02] G. Laporte, J.A., Mesa, and F.A. Ortega. Locating stations on rapid transit lines. *Computers & Operations Research*, 29(6):741–759, 2002.

[LLER11] R. M. Lusby, J. Larsen, M. Ehrgott, and D. Ryan. Railway track allocation: models and methods. *OR spectrum*, 33(4):843–883, 2011.

[LM07] C. Liebchen and R.H. Möhring. The modeling power of the periodic event scheduling problem: railway timetablesâĂŤand beyond. In *Algorithmic methods for railway optimization*, pages 3–40. Springer, 2007.

[LMO00] G. Laporte, J.A. Mesa, and F.A. Ortega. Optimization methods for the planning of rapid transit systems. *European Journal of Operational Research*, 122(1):1–10, 2000.

[LMOP11] G. Laporte, J.A. Mesa, F.A. Ortega, and F. Perea. Planning rapid transit networks. *Socio-Economic Planning Sciences*, 45(3):95–104, 2011.

[LSS+10] C. Liebchen, M. Schachtebeck, A. Schöbel, S. Stiller, and A. Prigge. Computing delay resistant railway timetables. *Computers & Operations Research*, 37(5):857–868, 2010.

[Man80] C.E. Mandl. Evaluation and optimization of urban public transportation networks. *European Journal of Operational Research*, 5:396–404, 1980.

[Mar06] G. Maróti. *Operations research models for railway rolling stock planning*. PhD thesis, Technische Universiteit Eindhoven, 2006.

[MBP15] D. Makovsek, V. Benezech, and S. Perkins. Efficiency in railway operations and infrastructure management. 2015.

[MHS+14] J. Manitz, J. Harbering, M. Schmidt, T. Kneib, and A. Schöbel. Network-based source detection: From infectious disease spreading to train delay propagation. In *29th International Workshop on Statistical Modelling*, volume 1, pages 201–205, 2014.

[MHS+15] J. Manitz, J. Harbering, M. Schmidt, T. Kneib, and A. Schöbel. Network-based Source Detection for Propagation Processes on Complex Networks with Application to Train Delays on Railway Systems. *Journal of the Royal Statistical Society: Series C (JRSS-C)*, 2015. submitted.

[MKS+14] J. Manitz, T. Kneib, M. Schlather, D. Helbing, and D. Brockmann. Origin detection during food-borne disease outbreaks-a case study of the 2011 ehec/hus outbreak in germany. *PLoS currents*, 6, 2014.

[MOP13] J.A. Mesa, F.A. Ortega, and M.A. Pozo. A geometric model for an effective rescheduling after reducing service in public transportation systems. *Computers & Operations Research*, 40(3):737–746, 2013.

[MOP14] J.A. Mesa, F.A. Ortega, and M.A. Pozo. Locating optimal timetables and vehicle schedules in a transit line. *Annals of Operations Research*, 222(1):439–455, 2014.

[MoR13] Ministry of Railways. Year book 2011-12, 2013. Available online: http://www.indianrailways.gov.in/railwayboard/ Last access 20-10-2015.

[MS09] M. Michaelis and A. Schöbel. Integrating line planning, timetabling, and vehicle scheduling: a customer-oriented heuristic. *Public Transport*, 1(3):211–232, 2009.

[MSW06] S. Mecke, A. Schöbel, and D. Wagner. Stop location - complexity and approximation issues. In *5th workshop on algorithmic methods and models for optimization of railways*, number 06901 in Dagstuhl Seminar proceedings, 2006.

[MW03] A.T. Murray and X. Wu. Accessibility tradeoffs in public transit planning. *Journal of Geographical Systems*, 5(1):93–107, 2003.

[Nac98] K. Nachtigall. Periodic network optimization and fixed interval timetables. *Deutsches Zentrum für Luft–und Raumfahrt, Institut für Flugführung, Braunschweig*, 1998.

[NC15] B. Naveh and Contributors. JGraphT. see http://jgrapht.org/, 2015.

[OEC15] OECD/ITF. ITF Transport Outlook 2015. 2015.

[Opt07] Dash Optimization. Xpress-optimizer reference manual. *Dash Optimization Ltd., Englewood Cliffs, NJ*, 2007.

[Pat25] A. Patz. Die richtige auswahl von verkehrslinien bei großen strassenbahnnetzen. *Verkehrstechnik*, 50:51, 1925.

[SBP74] L.A. Silman, Z. Barzily, and U. Passy. Planning the route system for urban buses. *Computers & operations research*, 1(2):201–211, 1974.

[Sch01] A. Schöbel. A model for the delay management problem based on mixed-integer-programming. *Electronic Notes in Theoretical Computer Science*, 50(1):1–10, 2001.

[Sch05] A. Schöbel. Locating stops along bus or railway lines — a bicriteria problem. *Annals of Operations Research*, 136:211–227, 2005.

[Sch12] A. Schöbel. Line planning in public transportation: models and methods. *OR spectrum*, 34(3):491–510, 2012.

[Sch14] M. Schmidt. *Integrating routing decisions in network problems.* Springer, 2014.

[Seg74] M. Segal. The operator-scheduling problem: A network-flow approach. *Operations Research*, 22(4):808–823, 1974.

[Sev94] S.V. Sevast'janov. On some geometric methods in scheduling theory: a survey. *Discrete Applied Mathematics*, 55(1):59 – 82, 1994.

[SG13] M. Siebert and M. Goerigk. An experimental comparison of periodic timetabling models. *Computers & Operations Research*, 40(10):2251–2259, 2013.

[SHLW09] A. Schöbel, H.W. Hamacher, A. Liebers, and D. Wagner. The continuous stop location problem in public transportation. *Asia-Pacific Journal of Operational Research*, 26(1):13–30, 2009.

[Sim78] B. Simons. A fast algorithm for single processor scheduling. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 246–252, 1978.

[SS06] A. Schöbel and S. Scholl. Line planning with minimal traveling time. In *ATMOS 2005-5th Workshop on Algorithmic Methods and Models for Optimization of Railways*. Internationales Begegnungs-und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, 2006.

[SS10] M. Schachtebeck and A. Schöbel. To wait or not to wait-and who goes first? delay management with priority decisions. *Transportation Science*, 44(3):307–321, 2010.

[SU89] P. Serafini and W. Ukovich. A mathematical model for periodic scheduling problems. *SIAM Journal on Discrete Mathematics*, 2(4):550–581, 1989.

[TWB15] The World Bank. Rail lines (total route-km). website, 2015. Available online: http://data.worldbank.org/indicator/IS.RRS.TOTL.KM/countries. Last access 20-10-2015.

[Voo13] A.M. Voorhees. A general theory of traffic movement. *Transportation*, 40(6):1105–1116, 2013.

[Vuc81] V. Vuchic. *Urban Public Transportation: Systems and Technology*. Prentice-Hall, Englewood Cliffs, NJ, EU, 1981.

[WAC+14] M. Wardman, R. Anderson, B. Condry, N. Findlay, R. Brage-Ardao, and H. Li. Valuing convenience in public transport. 2014.

[YKAT13] A. Yamamura, M. Koresawa, S. Adachi, and N. Tomii. Identification of causes of delays in urban railways. *Computers in Railways XIII: Computer System Design and Operation in the Railway and Other Transit Systems*, 127:403, 2013.

Addendum A

E. Carrizosa, J. Harbering, A. Schöbel
Minimizing the passengers' traveling time in the stop location problem
Journal of Operations Research

Minmizing the passengers' taveling time in the stop location problem

Emilio Carrizosa *

Universidad de Sevilla

C/ Tarfia s/n, 41012 Sevilla, Spain

ecarrizosa@us.es

Jonas Harbering †

Georg-August Universität Göttingen

Lotzestraße 16-18, 37083 Göttingen, Germany

jo.harbering@math.uni-goettingen.de

Anita Schöbel †

Georg-August Universität Göttingen

Lotzestraße 16-18, 37083 Göttingen, Germany

schoebel@math.uni-goettingen.de

December 11, 2015

In this paper we consider the location of stops along the edges of an already existing public transportation network. The positive effect of new stops is given by the better access of the passengers to the public transport network, while the passengers' traveling time increases due to the additional stopping activities of the trains which is a negative effect for the passengers.

The problem has been treated in the literature where the most common model is to cover all demand points with a minimal number of new stops.

*The first author is partially supported by MTM2012-36163-C06-03 and P11-FQM-7603.

†The other authors are partially supported by the European Union Seventh Framework Programme (FP7-PEOPLE-2009-IRSES) under grant number 246647 and the New Zealand Government (project OptALI).

In this paper, we follow this line and seek for a set of new stations covering all demand points but instead of minimizing the number of new stops we minimize the additional passengers' traveling time due to the new stops. For computing this additional traveling time we take not only the stopping times of the vehicles but also acceleration and deceleration of the vehicles into account.

We show that the problem is NP-hard, but we are able to derive a finite candidate set and two tractable IP formulations. For linear networks we show that the problem is polynomially solvable. We also discuss the differences to the common models from literature showing that minimizing the number of new stops does not necessarily lead to a solution with minimal additional traveling times for the passengers. We finally provide a case study showing that our new model decreases the traveling times for the passengers while still achieving the minimal number of new stops.

1 Introduction

The acceptance of public transportation depends on various components, among them convenience, punctuality, and reliability. In this paper, we address the question of convenience for the passengers. In particular, we investigate the problem of establishing additional stops (or stations) in a given public transportation network. The goal is, on the one hand, to improve the accessibility to the transportation network, but on the other hand not to increase the traveling time of passengers too much.

Due to their great potential for improving public transportation systems, several versions of the *stop location problem* (also called *station location problem*) have been considered in the literature. In order to find "good" locations for new stops, several objective functions are possible. One of the most frequently discussed goals is to minimize the number of stops such that each demand point is *covered*, i.e., it is within a tolerable distance from at least one stop. The maximal distance r that a passenger is willing to tolerate is called *covering radius*. For bus stops a covering radius of 400 m is common. In rail transportation, the covering radius is larger, often 2 km are used.

In the literature, *discrete* stop location problems have been introduced in [Gle75] and considered in [MDSF98, Mur01, LMO02, Mur03, WM05], see also references therein. In these papers, a finite set of potential stops is given. The goal is to choose a minimal

number of stops from this set such that each demand point is covered. In contrast to this discrete setting, [HLS+01] are the first to allow a *continuous* set of possible locations for the stops, for instance, all points on the current bus routes or railway tracks. This is motivated by a real-world application within a project with the largest German rail company (Deutsche Bahn). The authors consider the trade-off between the positive and negative effects of stops. The negative effect of longer passengers' traveling times due to additional stops is compared with the positive effect of shorter access times. Based on this application, the continuous stop location problem has also been treated theoretically in the literature. The goal is to locate a minimal number of stops along railway lines to cover a set of given demand points in the plane. Variants of this problem have been studied in [KPS+03, Sch05, Sch06, SHLW09]. The problem has been solved for the case of two intersecting lines, see [MMW04]. Algorithmic approaches for solving the underlying covering problem have been studied in [MW04, RS04]. Complexity and approximation issues have been presented in [MSW06].

A different objective function is to minimize the sum of distances from the passengers to the public transportation system, i.e., the sum of the distances between the demand facilities and their closest stops, see [MW03, PHHS09]. Recently, covering a set of OD-pairs with a given number of stops has been studied, see e.g., [KMP+12] and references therein.

Research done so far mostly deals with minimizing the number of new stops. This can be seen as a rough approximation of the passengers' traveling time, namely adding a fixed time penalty (often assumed to be two minutes) for each stop, see [SHLW09]. Since trains have a long acceleration and deceleration phase such an approximation is unrealistic in practice, in particular in metropolitan and regional transportation networks. In this paper we do not minimize the number of new stops but consider stop location problems minimizing the passengers' traveling time while taking the realistic vehicles' driving times including acceleration and deceleration into account. Note that we implicitly assume that there is no interference with other transportation modes that limits the speed of the vehicles, but that the trains can accelerate to their planned cruising speed without any restriction.

The remainder of the paper is structured as follows. We introduce the realistic driving time function and the stop location problem based on this function in Section 2. In Section 3 we compare our new model with the model from [SHLW09] and derive a finite dominating set in Section 4. Proposals for solving our new model are given in Section 5. Its complexity is analyzed and two different integer programming formulations are given.

Moreover, we show that the problem along one single line is polynomially solvable. In Section 6 we present a case study with numerical results on the quality and solvability of our new model.

2 Models for continuous stop location

In this section we first repeat the continuous stop location model common in the literature: Cover all demand points with a minimal number of new stops. We then introduce our new objective function in which we minimize the traveling time of the passengers instead of the number of the new stops.

Basic definitions: Locating stops along the edges of a network

The continuous stop location problem has been treated in the literature. We repeat its basic definitions.

Let $G = (V, E)$ a the given railway network with existing stops V and direct connections E between these stops. For each edge $e = (i, j) \in E$ an edge length $d_e \geq 0$ and a number of passengers $w_e \geq 0$ traveling along edge e is given.

A *point* in G is given as $s = (e, x) \in e$, i.e., it is defined by the edge $e = (i, j)$ and its distance $d(i, s) = x$ to the start vertex of e. The distance from x to the end vertex of e hence is $d(s, j) = d_e - x$, $0 \leq x \leq d_e$. Note that $i = (e, 0)$ and $j = (e, d_e)$. The set of points of G is denoted as $\mathcal{S} = \bigcup_{e \in E} e$. The set of points between two points $s_1 = (e, x_1)$ and $s_2 = (e, x_2)$ on the same edge is denoted as $[s_1, s_2] = \{(e, x) : x_1 \leq x \leq x_2\}$.
A new stop s in the network may be any point $s = (e, x)$. In the continuous stop location problem we want to identify a set of points in $\mathcal{S}$ as new stops, see Figure 1 as an example of a network with six existing and four new stops.

The constraints: covering all demand points

As in the common models in the literature we require that all (potential) passengers live close enough to at least one station. To this end, we assume that a finite set $\mathcal{P} \subseteq \mathbb{R}^2$ of demand points is given, that the railway network $G = (V, E)$ is embedded in the plane and that the access times from the demand points to the railway network can be measured by a distance function $dist : \mathbb{R}^2 \text{time} \mathbb{R}^2 \to \mathbb{R}$ which has been derived from a norm, i.e.,

$$dist(x, y) = \|y - x\|$$

for a given norm $\|\cdot\|$.

Definition 2.1. Let a covering radius $r \geq 0$ be given. For a set $S \subseteq \mathcal{S}$ and a set of demand points $\mathcal{P}$ we define: A demand point $p \in \mathcal{P}$ is *covered* by S if

$$d(p, s) \leq r \text{ for some } s \in S.$$

The *cover* of S is given as

$$\text{cover}(S) = \{p \in \mathcal{P} : dist(p, s) \leq r \text{ for some } s \in S\}.$$

If $\text{cover}(S) = \mathcal{P}$ we say that $\mathcal{P}$ *is covered* by S.

The goal is to find a set of stops $S \subseteq \mathcal{S}$ such that all demand points are covered, i.e., with $\text{cover}(S \cup V) = \mathcal{P}$.

Since we assumed that all elements of V are stops, demand points p which are covered by vertices $v \in V$ need not be considered. We hence may assume that

$$\mathcal{P} \subseteq \mathbb{R}^2 \setminus \text{cover}(V). \tag{1}$$

The stop location problem in the literature: Minimizing the number of new stops

In the literature, the following stop location problem (SL) has been considered: One looks for a set of stops $S^* \subseteq \mathcal{S}$ with minimal size covering all demand points:

(SL) Let $G = (V, E)$ be a graph and a finite set of points $\mathcal{P} \subseteq \mathbb{R}^2$ be given. Find a subset $S^* \subseteq \mathcal{S}$, such that $\text{cover}(S^*) = \mathcal{P}$ and $\lvert S^* \rvert$ is minimized.

Our new objective function: Minimizing the passengers' traveling times

In the stop location problems considered in the literature so far, the traveling times for passengers due to new stops can be estimated by adding a penalty time_{pen} for every stop to be located. This is an exact estimate if the distance between two stops is larger than the distance needed for acceleration and deceleration, and if time_{pen} gives the complete loss of vehicles' driving time resulting from the additional stop, i.e., the loss resulting from waiting t minutes at the station to allow passengers to board and deboard and the loss resulting from decelerating and accelerating instead of going by full speed. As an example, time_{pen} is estimated as two minutes for German regional trains (see [HLS+01]). However,

since trains accelerate slowly, this estimate is not realistic if the distance between two stops is rather short.

In order to consider acceleration and deceleration in stop location problems we first introduce a function describing the driving time of a train between two consecutive stops. This function depends on the distance d between those two consecutive stops. It is a simple consequence from Newton's laws of motion and has been used, e.g., in in [KWH75, Vuc81, DDW09].

Lemma 2.2. *Let a maximum cruising speed $v_0 > 0$, an acceleration of $a_0 > 0$ and a deceleration of $b_0 > 0$ of a vehicle be given. Then the vehicles' driving time for a distance d between two consecutive stops is given as*

$$T(d) = \begin{cases} \sqrt{\frac{2(a_0+b_0)}{a_0 b_0} d} & \text{if } d \leq d^{max}_{v_0,a_0,b_0} \\ \frac{d}{v_0} + \frac{v_0}{2a_0} + \frac{v_0}{2b_0} & \text{if } d \geq d^{max}_{v_0,a_0,b_0} \end{cases} \quad \text{where} \quad d^{max}_{v_0,a_0,b_0} = \frac{v_0^2}{2a_0} + \frac{v_0^2}{2b_0}$$

In [KWH75] the driving time function for the case $a_0 = b_0$ is used and its practical relevance is analyzed for fire engines in New York City.

Note that $d^{max}_{v_0,a_0,b_0}$ is the point where the driving time function turns from a square root behavior to a linear behavior. The shape and exact values of the function can be easily calculated; its main properties can be verified straightforwardly:

Lemma 2.3. *$T(d)$ is continuous, differentiable, concave, subadditive and monotonically increasing. Furthermore, for any d we have $\sqrt{\frac{2(a_0+b_0)}{a_0 b_0} d} \leq \frac{d}{v_0} + \frac{v_0}{2a_0} + \frac{v_0}{2b_0}$.*

A proof of these properties can be found, e.g., in [DDW09].

We now can derive the passengers' traveling times which depend on the new stops $S \subseteq \mathcal{S}$ to be located. In order to do so, we need to refine the railway network according to the new stops S. This is formalized next.

Given a finite set $S \subseteq \mathcal{S}$ of points of G, every set

$$S_e = S \cap e = \{s_1, \ldots, s_p\} \subseteq e$$

of points on $e = (i, j)$ can be naturally ordered along the edge e such that $d(s_1, i) \leq \ldots \leq d(s_p, i)$, i.e., 'from left to right'. Let $\leq_e$ denote this ordering. Adding the $p = |S_e|$ points of S_e as new stops to the edge $e = (i, j)$ splits the edge e into $|S_e| + 1$ new edges

$$E'_e(S_e) = \{(i, s_1), (s_1, s_2), \ldots, (s_p, j)\}.$$

Defining

$$E'(S) = \bigcup_{e \in E} E'_e(S_e) \tag{2}$$

we receive a new network $(V \cup S, E'(S))$ which is a subdivision of the given network $G = (V, E)$ (see Figure 2). Note that every edge $e' \in E'(S)$ can be represented as $e' = ((e, x_i), (e, x_j))$, i.e., it belongs to one unique edge $e \in E$. For a new edge $e' = ((e, x_i), (e, x_j)) \in E'(S)$ we define:

- its length as $d_{e'} = |x_j - x_i|$,
- the number of passengers traveling along edge e' as $w_{e'} = w_e$.

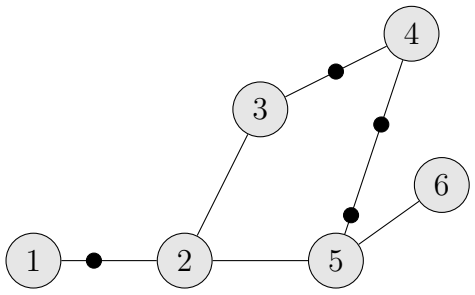


Figure 1: Locating $p = 4$ new stops on a Network $G = (V, E)$.

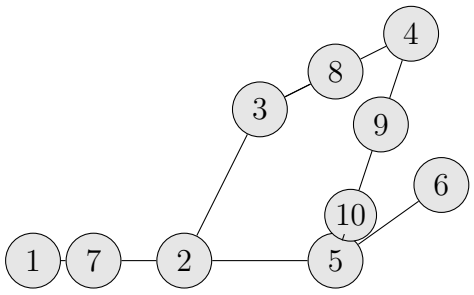


Figure 2: The new network $(V \cup S, E(S))$ with $p = 4$ new stops.

Given a set S of points on the graph G, we can finally define the *passengers' traveling time function* as

$$g(S) := \sum_{e \in E} w_e \left(|S_e| t + \sum_{e' \in E'_e(S_e)} T(d_{e'}) \right) = \sum_{e \in E} w_e |S_e| t + \sum_{e' \in E'(S)} w_{e'} T(d_{e'}). \tag{3}$$

Note that this function depends on the parameters $v_0 > 0$, $a_0 > 0$ and $b_0 > 0$ which model the vehicles' behavior in the vehicles driving time function T.
In this formula, we sum over the traveling times on all edges. The first term, $|S_e|t$, refers to the time which is given by stopping t minutes at every additional station on edge $e \in E$. Note that the stopping times at the already existing stops $i \in V$ are independent of the location of the new stops and hence neglected in the formula). The second term, $\sum_{e' \in E'_e(S_e)} T(d_{e'})$, accounts for the vehicles driving time on edge e including acceleration and deceleration at every new stop in S_e. Both values are multiplied by the number of passengers w_e on edge e to obtain the passengers' traveling time along this edge. Note

that the *additional* traveling time for the passengers is given as

$$f(S) = g(S) - g(\emptyset).$$

Note that in the case that no new station is built along an edge $e \in E$ we have that $S_e = \emptyset$ and $E'_e(S_e) = \{e\}$ and hence obtain

$$|S_e|t + \sum_{e' \in E'_e(S_e)} T(d_{e'}) = T(d_e).$$

The stop location problem minimizing passengers' traveling time

We summarize our new model: covering all demand points with a set of stops S such that the passengers' traveling time function $g(S)$ is minimal:

(SL*) Let $G = (V, E)$ be a graph with edge weights w_e for all $e \in E$, $\mathcal{P} \subseteq \mathbb{R}^2$ be a finite set of points, and $t \geq 0$. Moreover, let $v_0 > 0$, $a_0 > 0$ and $b_0 > 0$ be the parameters for the vehicles driving time.
Find a subset $S^* \subseteq \mathcal{S}$, such that $cover(S^*) = \mathcal{P}$ and $g(S^*)$ is minimized.

3 Comparing (SL) and (SL*)

(SL) minimizes the number of new stops while (SL*) considers the traveling times for the passengers. We start by showing that these are, in fact, different objective functions: Our example shows a case in in which the passengers' traveling time can be reduced by building two stops instead of only one.

Example 3.1. In Figure 3 two demand points p_1 and p_2 have to be covered by stops on $e = (v_1, v_2)$. As indicated in the figure, $d(p_1, v_1)$ and $d(p_2, v_2)$ is only a little bit larger than the covering radius r, and both demand points can be covered from the midpoint of the edge e. For this example let $w_e = 1$.

In order to minimize the number of stops it is sufficient to build only one stop, namely the midpoint of e, s_2. I.e., $S = \{s_2\}$ is an optimal solution to (SL). We compare S with the solution $\tilde{S} = \{s_1, s_3\}$, where s_1 and s_3 are the leftmost and rightmost points on e from which we can cover p_1 and p_2. Given some $\varepsilon < d^{max}_{v_0,a_0,b_0}$ the points s_1, s_2, and s_3 can be constructed such that $d(v_1, s_1) = d(v_2, s_3) = \varepsilon$ and such that $d_{v_1,s_2}, d_{s_2,v_2} \geq d^{max}_{v_0,a_0,b_0}$.

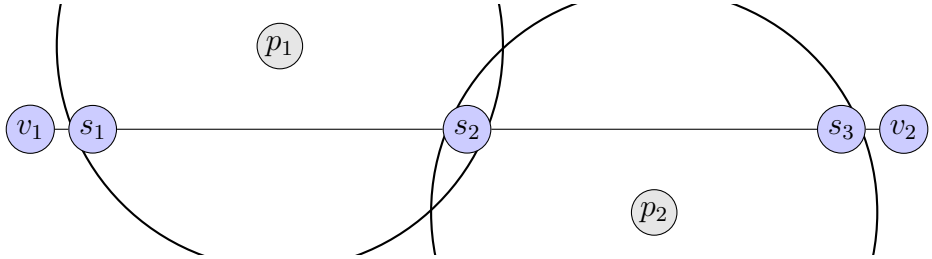


Figure 3: Example in which an optimal solution to (SL*) has more stops than an optimal solution to (SL).

Then the passengers' traveling times can be computed as

$$\begin{aligned} g(S) &= t + T(d_{v_1,s_2}) + T(d_{s_2,v_2}) = t + \tfrac{d_e}{v_0} + \tfrac{v_0}{a_0} + \tfrac{v_0}{b_0} \\ g(\tilde{S}) &= 2t + T(d_{v_1,s_1}) + T(d_{s_1,s_3}) + T(d_{s_3,v_2}) \\ &= 2t + 2\sqrt{\tfrac{2(a_0+b_0)}{a_0 b_0}\varepsilon} + \tfrac{d_e - 2\varepsilon}{v_0} + \tfrac{v_0}{2a_0} + \tfrac{v_0}{2b_0} \end{aligned}$$

and by letting ε tend to 0, we see that the relation between a_0, b_0 and t determines whether $g(\tilde{S}) < g(S)$. We obtain that $g(\tilde{S}) < g(S)$ if $\frac{v_0}{2a_0} + \frac{v_0}{2b_0} > t$, i.e., if breaking and accelerating takes longer than halting.

Even more, examples of the same pattern can be constructed which show that the number of stops in an optimal solution to (SL*) can differ by an arbitrarily large number from the number of stops in an optimal solution to (SL).

It is trivial that in (SL) more stops increase the value of the objective function. The previous example shows that this is not the case for (SL*), the objective function g is better for the two stops s_1 and s_3 as for the single stop s_2. However, adding an *additional* stop to a set of stops S always worsens the objective function even in (SL*):

Lemma 3.2. *Let $S^1 \subseteq S^2$ be two sets of points. Then $g(S^1) \leq g(S^2)$.*

Proof. We show that adding a single new station increases the traveling time, i.e., that $g(S^1) \leq g(S^2)$ for $S^2 = S^1 \cup \{s\}$. If $s \in S^1$ there is nothing to show. Hence, let $s \notin S^1$, and let $s = (e, x)$ for some edge $e = (i, j) \in E$. Clearly, $|S^1_e| = |S^1 \cap e| \leq |S^2 \cap e| = |S^2_e|$. Moreover, let $s_a = (e, x_a) \leq_e s \leq_e s_b = (e, x_b)$ for $s_a, s_b \in S^1_e \cup \{i, j\}$ being the neighbors of s in S^1_e. Then we have

$$T(x_b - x_a) \leq T(x_b - x) + T(x - x_a)$$

due to the subadditivity of T (see Lemma 2.3), and since $w_e \geq 0$ the result follows. □

In the next lemma we show that the two models are equivalent if we require a minimal distance of $d^{max}_{v_0,a_0,b_0}$ between any two new stops and between any new stop and an existing stop $v \in V$.

Lemma 3.3. *(SL*) with $w_e = 1$ for all $e \in E$ and with the additional constraints*

$$\begin{aligned} d(s,s') &\geq d^{max}_{v_0,a_0,b_0} \quad \textit{for all } s,s' \in S \\ d(s,v) &\geq d^{max}_{v_0,a_0,b_0} \quad \textit{for all } s \in S, v \in V \end{aligned}$$

is equivalent to (SL).

Proof. We compute the objective function g using the assumptions of the lemma and obtain

$$\begin{aligned} g(S) &= t\sum_{e\in E}|S_e| + \sum_{e\in E}\sum_{e'\in E'_e(S_e)} T(d_{e'}) \\ &= t|S| + \sum_{e\in E}\sum_{e'\in E'_e(S_e)} \tfrac{d_{e'}}{v_0} + \tfrac{v_0}{2a_0} + \tfrac{v_0}{2b_0} \\ &= t|S| + \sum_{e\in E} \tfrac{d_e v_0}{+} \sum_{e\in E}(|S_e|+1)\left(\tfrac{v_0}{2a_0} + \tfrac{v_0}{2b_0}\right) \\ &= \left(t + \tfrac{v_0}{2a_0} + \tfrac{v_0}{2b_0}\right)|S| + const, \end{aligned}$$

i.e., is equivalent to minimizing the number $|S|$ of new stops. □

In Theorem 4.6 in Section 4.2 we will provide a special case in which the additional constraints of the previous lemma are always satisfied, i.e., in which (SL) and (SL*) are equivalent.

Note that a numerical comparison between the solutions provided by (SL) and those provided by (SL*) will be presented within a case study in Section 6.

4 Feasibility and a finite candidate set for (SL*)

In this section we analyze the problem (SL*). We discuss its feasibility and provide a finite candidate set for (SL*). Let us start with the feasibility of (SL*).

4.1 Feasibility

(SL*) need not be feasible, but if it is it admits a finite solution whose objective value can be bounded.

Lemma 4.1.

- *(SL*) has a solution if and only if* $cover_{\mathcal{P}}(\mathcal{S}) = \mathcal{P}$.
- *If (SL*) has a feasible solution, then it also has a finite solution and*

$$g(S^*) \leq \max_{e \in E} w_e \left(|\mathcal{P}|t + (|E| + |\mathcal{P}|) \left(\max_{e \in E} \frac{d_e}{v_0} + \frac{v_0}{2a_0} + \frac{v_0}{2b_0} \right) \right)$$

Proof. The first part of the lemma is obvious. For the second part, let (SL*) be feasible. Then there exists a point $s_p \in \mathcal{S}$ such that $dist(p, s_p) \leq r$ for every demand point $p \in \mathcal{P}$. Choose $S := \{s_p : p \in \mathcal{P}\}$ as a feasible solution, i.e. $|S| \leq |\mathcal{P}|$. Each stop $s \in S$ adds at most one new edge to $E'(S)$, hence $|E'(S)| \leq |E| + |\mathcal{P}|$. Let $e' = (i, j) \in E'(S)$ be a new edge with $i = (\bar{e}, x_i), j = (\bar{e}, x_j)$ for some $\bar{e} \in E$. Then we estimate $d_{e'} \leq d_{\bar{e}} \leq \max_{e \in E} d_e$, and since T is monotone we obtain

$$T(d_{e'}) \leq \max_{e \in E} T(d_e) \leq \max_{e \in E} \frac{d_e}{v_0} + \frac{v_0}{2a_0} + \frac{v_0}{2b_0},$$

where the second inequality holds due to Lemma 2.3. Hence,

$$\begin{aligned} g(S^*) \leq g(S) &\leq \max_{e \in E} w_e \sum_{e \in E} \left(|S_e|t + \sum_{e' \in E'(S_e)} T(d_{e'}) \right) \\ &\leq \max_{e \in E} w_e \left(|\mathcal{P}|t + (|E| + |\mathcal{P}|) \left(\max_{e \in E} \frac{d_e}{v_0} + \frac{v_0}{2a_0} + \frac{v_0}{2b_0} \right) \right) \end{aligned}$$

and the result follows. □

This shows that feasibility of a solution is easy to check.

4.2 A finite dominating set for (SL*)

In the following we show that (SL*) can be reduced to a discrete problem by identifying a finite dominating set, i.e., a finite set of candidates $\mathcal{S}_{cand} \subseteq \mathcal{S}$, for which we know that it contains an optimal solution S^* if the problem is feasible. Such a finite dominating set will enable us to derive an IP formulation in Section 5.2. It turns out that we can use the same finite dominating set which has been used as candidate set for solving (SL) (see

[SHLW09]). Throughout this section, let us assume that (SL*) is feasible, which can be tested (due to Lemma 4.1).
For an edge $e = (i, j) \in E$ we define

$$T^e(p) = \{s \in e : dist(p, s) \leq r\}$$

as the set of all points on the edge $e \subseteq \mathcal{S}$ that can be used to cover demand point p.
Since $T^e(p) = e \cap \{x \in \mathbb{R}^2 : dist(p, x) \leq r\}$ is the intersection of two convex sets, and contained in e, it turns out to be a line segment itself. This observation is due to [SHLW09].

Lemma 4.2 ([SHLW09])**.** *For each demand point $p \in \mathbb{R}^2$ the set $T^e(p)$ is either empty or an interval contained in edge e.*

Let f_p^e, l_p^e denote the endpoints of the interval $T^e(p)$ We write $[f_p^e, l_p^e] = T^e(p)$.
For each edge $e = (i, j)$ we define

$$\mathcal{S}_{cand}^e := \bigcup_{p \in \mathcal{P}} \{f_p^e, l_p^e\} \cup \{i, j\},$$

which can be ordered along the edge e with respect to $\leq_e$. Let the resulting set be given as $\mathcal{S}_{cand}^e = \{s_0, s_1, \ldots, s_{N_e}\}$. In the following we show that

$$\mathcal{S}_{cand} = \bigcup_{e \in E} \mathcal{S}_{cand}^e$$

is a finite dominating set for (SL*).

From [SHLW09] we know that moving a point $s \in \mathcal{S}$ until it reaches an element of $\mathcal{S}_{cand}$ does not change $cover_{\mathcal{P}}(\{s\})$.

Lemma 4.3 ([SHLW09])**.** *Let $s \in e$ for an edge e of E, and let $s_j, s_{j+1} \in \mathcal{S}_{cand}$ be two consecutive elements of the finite dominating set with $s_j <_e s <_e s_{j+1}$. Then*

$$cover_{\mathcal{P}}(\{s\}) \subseteq cover_{\mathcal{P}}(\{s_j\}) \cap cover_{\mathcal{P}}(\{s_{j+1}\}),$$

in particular, the cover of s does not decrease when moving s between s_j and s_{j+1}.

Now we are able to prove that $\mathcal{S}_{cand} = \bigcup_{e \in E} \mathcal{S}_{cand}^e$ is, indeed, a finite dominating set for (SL*).

Theorem 4.4. *Either (SL*) is infeasible, or there exists an optimal solution $S^* \subseteq \mathcal{S}_{cand} \backslash V$.*

Proof. Let $S^* \subseteq \bigcup_{e \in E, p \in \mathcal{P}} T^e(p)$ be a feasible solution. We first show that $S^* \subseteq \mathcal{S}_{cand}$.

Assume, no. Our goal is to replace each $\tilde{s} \in S^* \setminus \mathcal{S}_{cand}$ by a point in $\mathcal{S}_{cand}$ without loosing feasibility and without worsening the objective function. To this end, take some $\tilde{s} \in S^* \setminus \mathcal{S}_{cand}$. If $\tilde{s} \in V$ there is nothing to show. Otherwise $\tilde{s} = (e, x) \in E$. Now find the following points on edge e:

- $s_j = (e, x_j), s_{j+1} = (e, x_{j+1}) \in \mathcal{S}_{cand}$ with $s_j <_e \tilde{s} <_e s_{j+1}$ for two consecutive elements of $\mathcal{S}^e_{cand}$ (which exist on e), and
- moreover find a point $s_{left} \in e$ such that $s_{left} \in (S^* \cup V) \cap e$, $s_{left} <_e \tilde{s}$ and no other point $s' \in (S^* \cup V) \cap e$ exists with $s_{left} <_e s' <_e \tilde{s}$.
  Analogously find $s_{right} \in e$ such that $s_{right} \in (S^* \cup V) \cap e$, $\tilde{s} <_e \tilde{s_{right}}$ and no other point $s' \in (S^* \cup V) \cap e$ exists with $\tilde{s} <_e s' <_e s_{right}$.

We now investigate the objective function if we move $\tilde{s}$. For all $s = (e, x)$ with $s_{left} <_e s <_e s_{right}$ the objective function $h(x) := g(S \setminus \tilde{s} \cup \{(e, x)\})$ is given as

$$h(x) = const + w_e \left(T(x - x_{left}) + T(x_{right} - x)\right)$$

where the constant part is independent of the choice of x in $s = (e, x)$, since x only influences the subedges (s_{left}, s) and (s, s_{right}) and leaves all other parts of the objective function untouched.
We hence have the following two properties:

(1) Since the composition of a concave and a linear function is concave, and the sum of two concave functions is also concave, we obtain that $h(x)$ is concave in x on the segment between s_{left} and s_{right}.

(2) From Lemma 4.3 we furthermore know that $cover_{\mathcal{P}}(\{s\}) \supseteq cover\{\tilde{s}\}$ for all $s = (e, x)$ between s_j and s_{j+1}, so we can shift $\tilde{s}$ between s_i and s_{i+1} without loosing feasibility.

Hence, due to the concavity (see (1)) of h, the minimization problem

$$\min\{h(x) = w_e \left(T(x - x_{left}) + T(x_{right} - x)\right) : \max\{x_{left}, x_j\} \leq x \leq \min\{x_{right}, x_{j+1}\}$$

has an optimal solution

$$x^* \in \{\max\{x_{left}, x_j\}, \min\{x_{right}, x_{j+1}\}\}$$

which is feasible for (SL*), see (2).

- In case that $x^* = x_j$ or $x^* = x_{j+1}$ we may replace $\tilde{s}$ by $s = (e, x^*) \in \mathcal{S}_{cand}$ and hence obtain a feasible solution with the same objective value.
- In case that $x^* = x_{left}$ or $x^* = x_{right}$ we may delete $\tilde{s}$ since the new solution is still feasible and improve the same objective value.

In both cases, we have reduced the number of points in $S^* \setminus \mathcal{S}_{cand}$. Proceeding with remaining points of S^* which do not belong to $\mathcal{S}_{cand}$ finally yields a feasible solution which is completely contained in $\mathcal{S}_{cand}$ and has the same (or a better) objective value as S^*.

In the proof we added the existing stops V to $\mathcal{S}_{cand}$. This has only been done for technical reasons. Since we know that no optimal solution contains a stop from V (since we assumed in (1) that $\mathcal{P} \subseteq \mathbb{R}^2 \setminus cover(V)$) we can also delete V from $\mathcal{S}_{cand}$. This finally finishes the proof. □

We hence have shown that (SL*) is equivalent to the following *discrete* problem:

(SL*-discrete) Let $G = (V, E)$ be a graph with edge weights w_e for all $e \in E$, $\mathcal{P} \subseteq \mathbb{R}^2$ be a finite set of points, and $t \geq 0$. Moreover, let $v_0 > 0$, $a_0 > 0$ and $b_0 > 0$ be the parameters for the vehicles driving time.
Find a subset $S^* \subseteq \mathcal{S}_{cand} \setminus V$, such that $cover(S^*) = \mathcal{P}$ and $g(S^*)$ is minimized.

We remark that there can be at most two candidates for each demand point on every edge, hence the total number of candidates $|\mathcal{S}_{cand}|$ is bounded by

$$|\mathcal{S}_{cand}| \leq 2|E||\mathcal{P}| + |V|.$$

Given the finite candidate set $\mathcal{S}_{cand}$ we define

$$\mathcal{E}_{cand} = \{c = (s, s') : s, s' \in \mathcal{S}_{cand} \text{ and } s, s' \in e \text{ for some edge } e \in E \text{ and } s \leq_e s'\}. \quad (4)$$

As usual, let d_c denote the length of $c \in \mathcal{E}_{cand}$. Note that

$$|\mathcal{E}_{cand}| = \sum_{e \in E} \binom{|\mathcal{S}^e_{cand}|}{2}.$$

The following corollary addresses the edges of the refined railway network $E(S^*)$ with new stations S^*, see (2).

Corollary 4.5. If (SL*) is feasible there exists an optimal solution S^* with $S^* \subseteq \mathcal{S}_{cand} \setminus V$ and $E(S^*) \subseteq \mathcal{E}_{cand}$.

As one consequence of Theorem 4.4 we will present a case in which (SL) and (SL*) are equivalent (as already promised in Section 3).

Theorem 4.6. *Let $d_c \geq d^{max}_{v_0,a_0,b_0}$ for all $c \in \mathcal{E}_{cand}$, and let $w_e = 1$ for all $e \in E$. Then (SL) and (SL*) are equivalent.*

Proof. Let S^* be an optimal solution to (SL*). Due to Theorem 4.4 we can assume without loss of generality that $S^* \subseteq \mathcal{S}_{cand} \setminus E$. From the assumption that $d_c \geq d^{max}_{v_0,a_0,b_0}$ for all $c \in \mathcal{E}_{cand}$ we know that

$$\begin{aligned} d(s,s') &\geq d^{max}_{v_0,a_0,b_0} \quad \text{for all } s,s' \in S^* \\ d(s,v) &\geq d^{max}_{v_0,a_0,b_0} \quad \text{for all } s \in S^* \text{ and for all } v \in V. \end{aligned}$$

Since $w_e = 1$ for all $e \in E$ we can use Lemma 3.3 and conclude that (SL) and (SL*) are equivalent. □

Theorem 4.4 has several other important consequences which will be exploited next: First, using the finite candidate set we can clarify the complexity status of (SL*) (see Section 5.1). Second, we can use the finite candidate set to derive IP formulations in Section 5.2. Finally, it also helps to solve the problem in polynomial time in a special case in Section 5.3.

5 Solving (SL*)

Using the finite candidate set $\mathcal{S}_{cand} \setminus V$ we can theoretically enumerate all potential solutions $S \subseteq \mathcal{S}_{cand} \setminus V$. However, this leads to a number of $\mathcal{O}\left(2^{2|E||\mathcal{P}|}\right)$ different solutions to be tested. In the following we hence discuss the complexity, IP formulations and a polynomially solvable case of (SL*-discrete).

5.1 Complexity of (SL*)

Having the finite candidate set we can now clarify the complexity status of (SL*).

Theorem 5.1. *(SL*) is NP-complete.*

Proof. To see that (SL*) is NP-hard, we reduce it to (SL) which is NP-hard according to [SHLW09]. To this end, we need another result of [SHLW09], namely that the finite candidate set $\mathcal{S}_{cand} \subseteq \mathcal{S}$ which we derived for (SL*) in Theorem 4.4 is also a finite candidate set for (SL). Let an instance of (SL) be given. We define the following instance of (SL*):

- We leave the network G and the demand points p as they are.
- We choose $w_e = 1$ for all $e \in E$.
- For choosing the parameters v_0, a_0 and b_0 we proceed as follows. We use the finite candidate set $\mathcal{S}_{cand}$ for (SL) and determine $\underline{m} := \min\{d(s,s') : s, s' \in \mathcal{S}_{cand}, \text{ and } s, s' \in e \text{ for some } e \in E\}$ as the closest distance between two candidate locations on the same edge. We then choose v_0, a_0, b_0 such that $d^{max}_{v_0,a_0,b_0} \leq \underline{m}$. (This is possible since $d^{max}_{v_0,a_0,b_0} \to 0$ if $a_0, b_0 \to \infty$).

We now claim that a solution to (SL) with $|S| \leq K$ exists if and only if a solution S^* to (SL*) exists with $g(S^*) \leq Kt + \sum_{e\in E} \frac{d_e}{v_0} + (|E| + K)\left(\frac{v_0}{2a_0} + \frac{v_0}{2b_0}\right)$.

To see this, first note, that (SL) is feasible if and only if (SL*) is feasible, and that optimal solutions of (SL) and (SL*) exist in case of feasibility. Furthermore, if $d_{e'} \geq d^{max}_{a_0,b_0,v_0}$ for all $e' \in E'(S)$ and $w_e = 1$ for all $e \in E$ the objective function of (SL*) becomes

$$\begin{aligned} g(S) &= |S|t + \sum_{e'\in E'(S)} \left(\frac{d_{e'}}{v_0} + \frac{v_0}{2a_0} + \frac{v_0}{2b_0}\right) \\ &= |S|t + \frac{1}{v_0} \sum_{e'\in E'(S)} d_{e'} + |E'(S)| \left(\frac{v_0}{2a_0} + \frac{v_0}{2b_0}\right) \qquad (5) \\ &= |S|t + \frac{1}{v_0} \sum_{e\in E} d_e + |E'(S)| \left(\frac{v_0}{2a_0} + \frac{v_0}{2b_0}\right) \end{aligned}$$

"⇒" Let S be an optimal solution to (SL) with $|S| \leq K$. Then there exists an optimal solution S^* to (SL) with $S^* \subseteq \mathcal{S}_{cand}$ according to the definition of $\mathcal{S}_{cand}$ (see above). S^* is the required solution to (SL*): Clearly, S^* is feasible for (SL*). Furthermore, $d_{e'} \geq \underline{m} \geq d^{max}_{a_0,b_0,v_0}$ for all $e' \in E'(S^*)$ (since $S^* \subseteq \mathcal{S}_{cand}$), and $|E'(S^*)| \leq |E| + K$. Hence $g(S^*) \leq |K|t + \sum_{e\in E} \frac{d_e}{v_0} + (|E| + K)\left(\frac{v_0}{2a_0} + \frac{v_0}{2b_0}\right)$.

"⇐" Let S^* be an optimal solution to (SL*) with $g(S^*) \leq |K|t + \sum_{e\in E} \frac{d_e}{v_0} + (|E| + K)\left(\frac{v_0}{2a_0} + \frac{v_0}{2b_0}\right)$. From Theorem 4.4 we know that there exists $S \subseteq \mathcal{S}_{cand}$ with $g(S^*) = g(S)$. We show that S is the required solution: Clearly, S is feasible. Since

$d_{e'} \geq \underline{m} \geq d^{max}_{a_0,b_0,v_0}$ for all $e' \in E(S^*)$ and $|E(S)| = |E| + |S|$ we have

$$|K|t + \sum_{e \in E} \frac{d_e}{v_0} + (|E| + K)\left(\frac{v_0}{2a_0} + \frac{v_0}{2b_0}\right) \geq g(S^*) = g(S) = |S|t + \sum_{e \in E} \frac{d_e}{v_0} + (|E| + |S|)\left(\frac{v_0}{2a_0} + \frac{v_0}{2b_0}\right)$$

which is equivalent to $|S| \leq K$.

□

As an immediate consequence we obtain:

Corollary 5.2. (SL*-discrete) is NP-complete.

Proof. This follows directly from Theorem 5.1 and Theorem 4.4. □

5.2 Integer programming formulations for (SL*)

In this section we develop two different valid IP-formulations for (SL*) and discuss their performance. Both IP-formulations are based on the candidate set developed in Section 4.2. We first define the *covering matrix* $A^{cov} = (a_{p,s})_{p \in \mathcal{P}, s \in \mathcal{S}_{cand}}$ by

$$a_{ps} = \begin{cases} 1 & \text{if } p \in cover_{\mathcal{P}}(\{s\}) \\ 0 & \text{otherwise.} \end{cases}$$

Then (SL) can be formulated as a covering problem (see, e.g., [Mur01]), namely

$$\text{(IP-SL)} \qquad \min \quad \sum_{s \in \mathcal{S}_{cand}} x_s \tag{6}$$

$$s.t. \quad \sum_{s \in \mathcal{S}_{cand}} a_{ps} x_s \geq 1 \qquad \forall\, p \in \mathcal{P} \tag{7}$$

where the variables x_s have the following meaning:

$$x_s = \begin{cases} 1 & \text{if stop } s \in \mathcal{S}_{cand} \text{ is built.} \\ 0 & \text{otherwise.} \end{cases}$$

We now develop two IP formulations for (SL*).

To this end, let $\geq_e$ denote the canonical ordering along edge $e \in E$, $\mathcal{E}_{cand}$ be defined as in (4), and let the length d_c and $T(d_c)$ be pre-calculated for all $c \in \mathcal{E}_{cand}$. Our first –

straightforward – IP formulation of the discrete version of (SL*) is given by a classical covering formulation in which we additionally require that all edges between consecutive stops must be built.

$$\text{(IP1-SL*)} \qquad \min \quad \sum_{e \in E} w_e \left(t \sum_{\substack{s \in \mathcal{S}_{cand} \\ s \in e}} x_s + \sum_{\substack{c \in \mathcal{E}_{cand} \\ c \in e}} T(d_c) y_c \right) \tag{8}$$

$$s.t. \quad \sum_{s \in \mathcal{S}_{cand}} a_{ps} x_s \geq 1 \qquad \forall\, p \in \mathcal{P} \tag{9}$$

$$x_{s_i} + x_{s_j} - \sum_{\substack{s \in [s_i, s_j] \cap \mathcal{S}_{cand} \\ s \notin \{s_i, s_j\}}} x_s \leq y_c + 1 \qquad \forall\, c = (s_i, s_j) \in \mathcal{E}_{cand} \tag{10}$$

$$x_s = 1 \qquad \forall\, s \in V \tag{11}$$

$$x \in \{0,1\}^{|\mathcal{S}_{cand}|} \tag{12}$$

$$y \in \{0,1\}^{|\mathcal{E}_{cand}|} \tag{13}$$

The variables x_s have the same meaning as before while the variables y_c have the following interpretation:

$$y_c = \begin{cases} 1 & \text{if edge } c \in \mathcal{E}_{cand} \text{ is built.} \\ 0 & \text{otherwise.} \end{cases}$$

Constraint (9) ensures that every demand point is covered as in (7). Constraints of type (10) ensure that a candidate edge is considered in the objective function if and only if it is built, i.e., if and only if its two endpoints are stops and no candidate between the two endpoints is also a stop. Recall that all $v \in V$ are considered as stops. This is ensured by constraint (11). Finally, the objective function (8) gives the passengers' traveling time:

Lemma 5.3. *The above stated IP formulation is correct for (SL*).*

Proof. It is obvious that a solution is only feasible if all demand points are covered. This is ensured by constraint (10) of the IP formulation. Furthermore we need to show that in any optimal solution a candidate edge c is built if and only if its two endpoints are chosen as stops and there is no other on c, i.e., that

$$\begin{aligned} & x_{s_i} = 1 \\ & x_{s_j} = 1 \quad \iff \quad y_c = 1 \text{ where } c = [s_i, s_j]. \\ x_{s_k} = 0 \quad & \forall i < k < j \end{aligned}$$

"⇒" Suppose $x_{s_i} = 1$, $x_{s_j} = 1$ and $x_s = 0$ for all $s \in [s_i, s_j] \cap \mathcal{S}_{cand} \setminus \{s_i, s_j\}$. Then $2 = x_{s_i} + x_{s_j} - \sum_{\substack{s \in [s_i,s_j] \cap \mathcal{S}_{cand} \\ s \notin \{s_i,s_j\}}} x_s \leq y_c + 1$ (since (10) is satisfied), i.e., we conclude that $y_c = 1$.

"⇐" Suppose we do not have $x_{s_i} = 1$, $x_{s_j} = 1$ and $x_{s_k} = 0$ for all $s_k \in [s_i, s_j]$ and $s_k \notin \{s_i, s_j\}$. Then $x_{s_i} + x_{s_j} - \sum_{\substack{s \in [s_i,s_j] \cap \mathcal{S}_{cand} \\ s \notin \{s_i,s_j\}}} x_s \leq 1$, i.e., $y_c = 0$ satisfies (10), hence, due to optimality $y_c \neq 1$.

□

The second IP-formulation is based on an interpretation of the problem as flow problem. It involves less constraints. In our numerical results we show that it is clearly superior to (IP1-SL*). The definition of the variables x and y of the IP-formulation remains the same as in (IP1-SL*). The IP-formulation (IP2-SL*) is given as

$$\text{(IP2-SL*)} \qquad \min \sum_{e \in E} w_e \left(t \sum_{\substack{s \in \mathcal{S}_{cand} \\ s \in e}} x_s + \sum_{\substack{c \in \mathcal{E}_{cand} \\ c \in e}} T(d_c) y_c \right) \tag{14}$$

$$s.t. \sum_{s \in \mathcal{S}_{cand}} a_{p,s} x_s \geq 1 \qquad \forall\, p \in \mathcal{P} \tag{15}$$

$$\sum_{\substack{s \in \mathcal{S}^e_{cand} \\ s <_e s_j}} y_{s,s_j} = x_{s_j} \quad \forall\, e \in E, \quad \forall\, s_j \in \mathcal{S}^e_{cand} \tag{16}$$

$$\sum_{\substack{s \in \mathcal{S}^e_{cand} \\ s_i <_e s}} y_{s_i,s} = x_{s_i} \quad \forall\, e \in E, \quad \forall\, s_i \in \mathcal{S}^e_{cand} \tag{17}$$

$$x_s = 1 \qquad \forall\, s \in V \tag{18}$$

$$x \in \{0,1\}^{|\mathcal{S}_{cand}|} \tag{19}$$

$$y \in \{0,1\}^{|\mathcal{E}_{cand}|} \tag{20}$$

Constraints (15) define the covering condition (see (7) in (IP-SL)). The flow understanding of the problem is expressed in constraints (16)–(17) which ensure that exactly one candidate edge is built on either side of a stop s with $x_s = 1$, and no candidate edge is built which ends at a stop s with $x_s = 0$. Finally, as in (IP1-SL*), constraint (18) ensures that all $v \in V$ are considered as stops.

Lemma 5.4. *(IP2-SL*) is a correct IP-formulation for (SL*).*

Proof. Let (x, y) be a feasible solution to (IP2-SL*). Assuming x to be fixed we may rewrite constraints (16) and (17) for every edge $e \in E$ to

$$\sum_{\substack{s' <_e s \\ s' \in \mathcal{S}^e_{cand}}} y_{s',s} = \sum_{\substack{s <_e s' \\ s' \in \mathcal{S}^e_{cand}}} y_{s,s'} \quad \text{for all } s \text{ with } x_s = 1, \text{ and} \tag{21}$$

$$\sum_{\substack{u <_e s' \\ s' \in \mathcal{S}^e_{cand}}} y_{u,s'} = 1 \tag{22}$$

$$\sum_{\substack{s' <_e v \\ s' \in \mathcal{S}^e_{cand}}} y_{s',v} = 1, \tag{23}$$

i.e., the y-variables describe a path passing exactly through the set of stops s with $x_s = 1$. We now show that a candidate edge $c = (s_i, s_j)$ is built if and only if its endpoints s_i, s_j are built and no station in between s_i and s_j is built.

"$\Leftarrow$": If s_i and s_j are built and no station between s_i and s_j are built then

$$\sum_{\substack{s_i <_e s' \\ s' \in \mathcal{S}^e_{cand}}} y_{s_i,s'} = 1$$

hence there exists a unique s with $y_{s_i,s} = 1$. If $s_i <_e s' <_e s_j$ then from (16) it follows $x_{s'} = 1$ which is a contradiction to the assumption that no station between s_i and s_j is built. If $s_j <_e s'$ then from (21) it follows that there exists a path from s_i to v. But since $x_{s_j} = 1$ and (21) there also exists a path (independent to the $s_i - v$-path due to (21)) from s_j to v. This is a contradiction to (23). Hence, $s' = s_j$ and $c = (s_i, s_j)$ is built.

"$\Rightarrow$": We now assume a candidate edge $c = (s_i, s_j)$ is built and show that, hence, the stops s_i and s_j are built and no station between s_i and s_j is built. Since c is built, from (16) we know that s_i is built and from (17) we know that s_j is built. From (21) it follows that s_j and v are connected by a path. Assume there was any $\tilde{s}$ such that $s_i <_e \tilde{s} <_e s_j$ and $x_{\tilde{s}} = 1$. From (21) we then now that there is also a path from $\tilde{s}$ to v (independent to the $s_i - v$-path due to (21)). Hence, we have a contradiction to (23) as the $s_i - v$-path and the $\tilde{s} - v$-path can not exist at the same time.

As before, (15) ensures that all demand points are covered. □

Note that any feasible solution to (IP2-SL*) consists of a set of paths in the network $(\mathcal{S}_{cand}, \mathcal{E}_{cand})$, namely exactly one path from u to v for any edge $e = (u, v)$ in the original

network $G = (V, E)$. Hence, (IP2-SL*) is similar to the *shortest covering path problem* as defined in [CPR94]. The differences are that the we seek for a set of shortest paths - one on each edge - such that the covering constraints are satisfied while in [CPR94] one single shortest path is sought. Another difference is that the demand points to be covered are contained in the plane and are not nodes of the network (as in [CPR94]); i.e., in our version we combine the discrete nature of choosing a potential solution with the geometric aspect of covering.

We now compare the sizes of the two different IP-formulations. The number of variables is the same for both formulations, namely $|\mathcal{S}_{cand}|+|\mathcal{E}_{cand}|$, but the number of constraints varies quite substantially. In the first formulation (IP1-SL*) we have $|\mathcal{P}| + |\mathcal{E}_{cand}|$ constraints which is of order $\mathcal{O}\left(|\mathcal{S}_{cand}|^2\right)$ while in (IP2-SL*) we have $|\mathcal{P}| + 2|\mathcal{S}_{cand}| + 2|V|$ constraints, i.e., of linear order $\mathcal{O}\left(|\mathcal{S}_{cand}|\right)$ only.

5.3 The special case (Line-SL*): Covering all demand points from a line

In this section we consider a special case in which (SL*) is polynomially solvable. Namely, we consider the problem (SL*) on a line, i.e., $V = \{u, v\}$ and $E = \{e\}$. To refer to the line structure of the underlying network G we call this problem (Line-SL*).

(Line-SL*) Let $G = (\{u, v\}, \{e\})$ be a (single-edge) graph, $\mathcal{P} \subseteq \mathbb{R}^2$ be a finite set of points and $t \geq 0$. Moreover, let $v_0 > 0$, $a_0 > 0$ and $b_0 > 0$ be the parameters for the vehicles driving time.
Find a subset $S^* \in \mathcal{S}$, such that $cover_{\mathcal{P}}(S^*) = \mathcal{P}$ and $g(S^*)$ is minimized.

Note that G only consists of one edge and hence there is only one weight w_e appearing in the objective function which in consequence has no effect on the solution and hence can be neglected.

As before, we will use the finite dominating set $\mathcal{S}_{cand}$ instead of the set of all points $\mathcal{S}$. Since only one edge is considered we can assume $\mathcal{S}_{cand}$ to be ordered with respect to $\leq_e$. This order has a nice property (see [HLS+01, Sch06]), namely, in the covering matrix A^{cov} the ones appear consecutively in every row. This property is called *consecutive-ones property* (C1P). It ensures that (SL) can be solved in polynomial time by linear programming. In [HLS+01, Sch06] a more efficient shortest path algorithm on a special graph is designed for solving (SL) on a line. In the following we transfer these results to our problem (Line-SL*),

i.e., to the case of minimizing the passengers' traveling times.
Let us define

$$l_p := \min\{j : s_j \in \mathcal{S}_{cand}, a_{p,s_j} = 1\} \text{ for } p \in \mathcal{P},$$
$$r_p := \max\{j : s_j \in \mathcal{S}_{cand}, a_{p,s_j} = 1\} \text{ for } p \in \mathcal{P}.$$

Recall that a matrix A^{cov} with (C1P) is called strictly monotone if $l_{p_1} < \ldots < l_{p_m}$ and $r_{p_1} < \ldots < r_{p_m}$ hold.

Lemma 5.5. *([Sch06]) Let A^{cov} define the covering matrix for an instance of (Line-SL*), then there exists an equivalent (Line-SL*) problem with covering matrix $\bar{A}^{cov}$, such that $\bar{A}^{cov}$ is a strictly monotone matrix.*

Thus, we can assume A^{cov} to be given as a strictly monotone matrix. Implicitly, this gives an ordering of $\mathcal{P}$ according to the ordering of the rows in the strictly monotone matrix. From the ordering also the minimum and the maximum are well defined. We will assume that $\mathcal{P} = \{p_1, \ldots, p_m\}$ is given in this ordering. Let us now define

$$\bar{l}_s := \min\{i : p_i \in \mathcal{P}, a_{p_i,s} = 1\} \text{ for } s \in \mathcal{S}_{cand},$$
$$\bar{r}_s := \max\{i : p_i \in \mathcal{P}, a_{p_i,s} = 1\} \text{ for } s \in \mathcal{S}_{cand}.$$

Now, we are able to construct a graph $\mathcal{N} = (\mathcal{V}, \mathcal{A})$ on which we will later solve a shortest path problem. Let

$$\mathcal{V} := \mathcal{S}_{cand}.$$

Note that $u, v \in \mathcal{S}_{cand}$. Furthermore, let

$$\mathcal{A} := \{(s_j, s_k) \ : \ s_j < s_k \text{ and } \bar{l}_{s_k} \leq \bar{r}_{s_j} + 1\} \cup \{(u, s_j) \ : \ \bar{l}_{s_j} = 1\} \cup \{(j, v) \ : \ \bar{r}_{s_j} = m\}.$$

Finally, we define costs for each edge $a = (s_j, s_k) \in \mathcal{A}$ as follows

$$cost_{s_j,s_k} = \begin{cases} T\left(d(s_j, s_k)\right) & \text{if } s_k = v \\ t + T\left(d(s_j, s_k)\right) & \text{else.} \end{cases}$$

$\mathcal{N}$ is a directed cycle-free graph. Let P be any $u - v-$path in $\mathcal{N}$. Then $P \subseteq \mathcal{S}_{cand}$ is uniquely described by its nodes, and its costs are given by $cost(P) = \sum_{a \in P} cost_a$. Based

on this notation the following theorem holds.

Theorem 5.6. *([Sch06]) Let $P \subseteq \mathcal{V}$. Then cover$(P) = \mathcal{P}$ if and only if P is an $u-v-$path in $\mathcal{N}$.*

With these results we can now provide a polynomial algorithm for solving (Line-SL*).

Theorem 5.7. *(Line-SL*) is solvable in polynomial time.*

Proof. The problem (Line-SL*) is solved as a shortest $u-v-$path problem on the respective graph $\mathcal{N}$. Let $P = \{u, s_{j_1}, \ldots, s_{j_p}, v\}$ be an optimal solution of the shortest $u - v-$path problem in $\mathcal{N}$. From Theorem 5.6 we obtain that P fulfills the covering constraint and thus is a feasible solution to (Line-SL*). We have to show that the solution is also optimal for (Line-SL*). Suppose no, i.e., there exists a solution $P^* = \{u, s^*_{j_1}, \ldots, s^*_{j_q}, v\}$ for (Line-SL*) with

$$g(P) > g(P^*).$$

Again using Theorem 5.6 we may interpret this solution as $u - v-$path in $\mathcal{N}$. This means

$$\begin{aligned}
\mathrm{C}ost(P) &= cost_{u,s_{j_1}} + \sum_{i=1}^{p-1} cost_{s_{j_i},s_{j_{i+1}}} + cost_{s_{j_p},v} \\
&= pt + T(d(u, s_{j_1})) + \sum_{i=1}^{p-1} T(d(s_{j_i}, s_{j_{i+1}})) + T(d(s_{j_p}, v)) \\
&> qt + T(d(u, s^*_{j_1})) + \sum_{i=1}^{q-1} T(d(s^*_{j_i}, s^*_{j_{i+1}})) + T(d(s^*_{j_q}, v)) \\
&= cost_{u,s^*_{j_1}} + \sum_{i=1}^{q-1} cost_{s^*_{j_i},s^*_{j_{i+1}}} + cost_{s^*_{j_q},v} \;=\; \mathrm{C}ost(P^*)
\end{aligned}$$

Hence, P^* defines a shorter $u - v-$path in $\mathcal{N}$ which is a contradiction to the optimality of P. The complexity applying Dijkstra's shortest path algorithm results in $\mathcal{O}(|\mathcal{P}|log|\mathcal{P}|)$, where O($|S_{cand}|$) and O($|\mathcal{P}|$) are of the same order since we consider a linear graph G. □

We conclude that simplifications of the general problem turn out to be solvable in polynomial time, while (SL*) is NP-hard due to Theorem 5.1.

6 Experiments

Environment. All our experiments were conducted on a PC with 24 six-core Intel Xenon X5650 Processor running at 2.67 GHz with 12 MB cache and a main memory of 94 GB. The IPs were solved using Xpress Optimizer v27.01.02. The running time limit of the solver was set to 300 seconds.

Benchmark sets. The southern part of the existing railway network of Lower Saxony, Germany, is used as the existing network $G = (V, E)$. From the same area the 34 largest cities are considered as demand points if they are not already close enough to an existing stop (see assumption (1)). This is the setting for the first benchmark set (LoSa=Lower Saxony).
In our second benchmark set we removed existing stops which have only two adjacent edges in order to obtain a more interesting set (LoSaRe=Lower Saxony Reduced) with longer tracks and more uncovered demand points. This set has higher complexity.
The values for the vehicles' driving times are chosen according to realistic properties, which are an acceleration and deceleration of 0.6 m/s^2, a cruising speed of 160 km/h and a stopping time t of $30s$. For a set of different radii ($r \in \{1500, 2000, 2500, \ldots, 15000\}$ (in meters)), we constructed instances containing all demand points which can be covered by r, i.e.,

$$\{p \in \mathcal{P} \setminus cover(V) : \text{ there exists } s \in \mathcal{S} : d(p, s) \leq r\}.$$

This means that $\mathcal{P}$ increases with the radius.

Setup. The quality and computation times when solving the IP formulations for (SL) and for (SL*) are compared. To this end, for each of the benchmark sets and every radius $r \in \{1500, 2000, 2500, \ldots, 15000\}$ both models have been solved by Xpress Optimizer. Then for each run the quality of the solution is measured by evaluating the passengers' traveling time and the number of stops.
If it is not explicitly stated, the IP-formulation (IP2-SL*) is used to solve (SL*) since it turned out to be more efficient, see Section 6.1.

Results. Table 1 summarizes our results calculating the average values of the additional traveling time for all instances. With additional traveling time for a solution S we mean the passengers traveling time $g(S)$ reduced by the traveling time which is not avoidable,

Instance (LoSa):	(SL)	(SL*)
passengers' traveling time $f(S)$	243.6	230.5
number of stops $\|S\|$	35.86	35.86

Instance (LoSaRe):	(SL)	(SL*)
passengers' traveling time $f(S)$	1019.6	998.6
number of stops $\|S\|$	17.64	17.64

Table 1: Average values of the additional traveling time for the solutions of (SL) and (SL*).

i.e., the traveling time $g(\emptyset)$ obtained when no additional stop is established:

$$f(S) = g(S) - g(\emptyset).$$

Note that minimizing the traveling time $g(S)$ and minimizing the additional traveling time $f(S)$ are equivalent problems, since $g(\emptyset)$ is a constant.

6.1 Computing times for solving (SL) and (SL*)

The generation of instances for different radii also allows to study the computation time for the IP-formulation of (SL) and for the two IP formulations (IP1-SL*) and (IP2-SL*) for (SL*). Note that all the IP formulations are based on the finite candidate set $\mathcal{S}_{cand}$.

In Figure 4 (lower part) we depict that the number of such candidates increases (linearly) with increasing radius. However, even a linear increase in $|\mathcal{S}_{cand}|$ leads to an exponential increase of the number of possible solutions $S \subset \mathcal{S}_{cand}$. We hence expect an exponential increase of the running time. Such an exponential increase is shown in the upper part of Figure 4, but only for solving (SL*) with the first IP formulation (IP1-SL*). (Since the maximal running time is set to 300 seconds we have a flat part at the end of the graph.)

The figure also clearly shows that the IP-formulations for (SL) and our second IP formulation (IP2-SL*) for (SL*) are good enough to be still able to solve the problem of bigger sizes in the same time (about 2 seconds). We conclude that (IP2-SL*) is much better than (IP1-SL*).
This justifies using the solutions of (IP2-SL*) for comparing the quality of the solutions in the following tests.

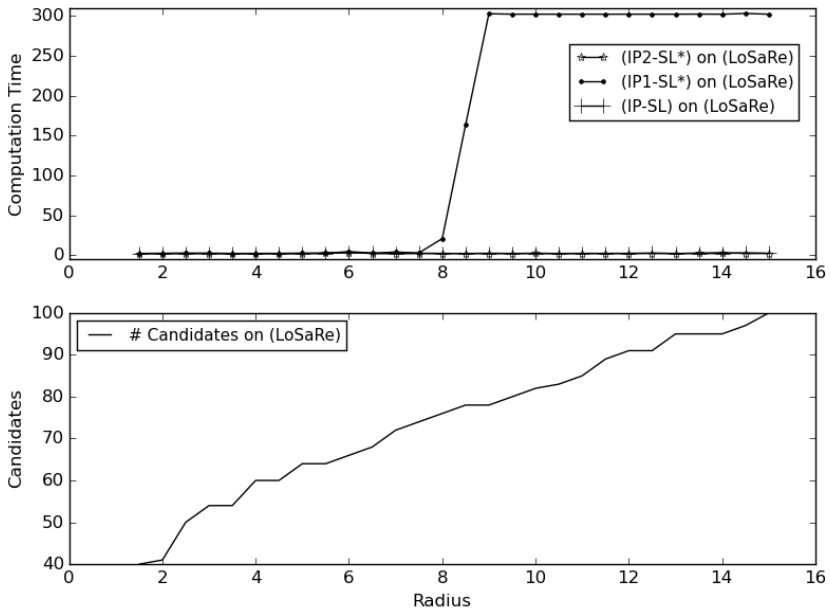


Figure 4: Computing time and number of candidates.

6.2 Comparison of objective function values of (SL) and (SL*)

(SL) minimizes the number of new stops and (SL*) minimizes the passengers' traveling time. We tested on our two benchmark sets (LoSa) and (LoSaRe) how big these differences are. The results are depicted in Figure 5.
We see that the solutions of (SL) and (SL*) in terms of the number of stops do not vary at all while in terms of the resulting additional passengers' traveling times (SL*) performs better than (SL).

- On the benchmark set (LoSa) the average additional passengers' traveling time can be reduced by $13s$ which amounts to an improvement of about 5.6% by using (SL*) instead of (SL). The maximal improvement on this instance reduces the additional passengers' traveling time by about $75s$ which are about 16.6%.

- On the instance (LoSaRe) we obtain an average time saving of $21s$ which result in an improvement of 2.1%. The maximal improvement achieves to save $78s$ which means an improvement of 5.1%. We conclude that on real-world instances (SL*) is not worse in the number of new stops but improves the additional passengers' traveling time.

Note that in terms of the absolute time saving the model (SL*) performs better on the

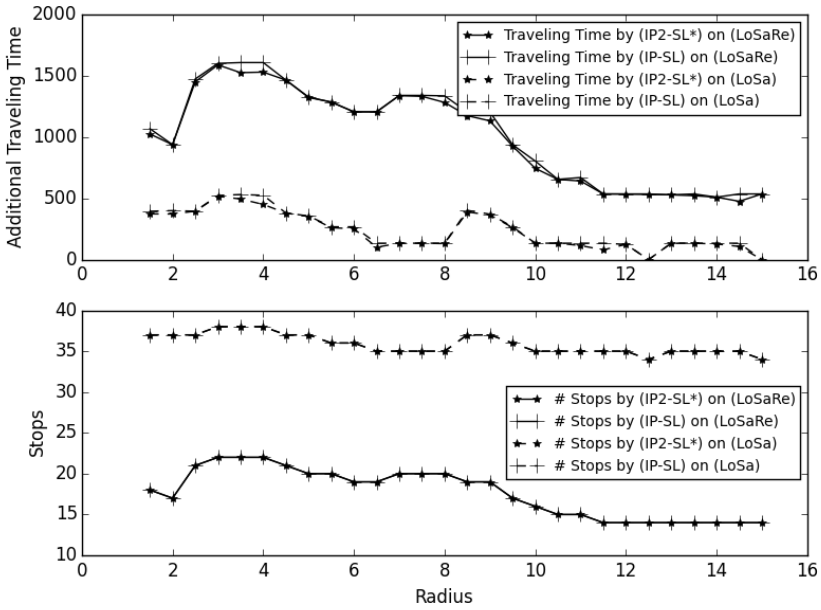


Figure 5: Passengers' traveling time and number of new stops.

network with longer edges, firstly because in (LoSaRe) more demand points have to be covered (less are uncovered), so more new stops are needed. Secondly, longer edges grant more freedom to do a bad allocation of the stops in terms of the traveling time (since the network structure remains the same for (LoSa) and (LoSaRe)).
We summarize that the relative improvement is higher on (LoSa) while the absolute improvement is larger on (LoSaRe).

We also looked at some other properties of our solutions.

Dependence on the covering radius. There is another interesting observation to be mentioned when looking at Figure 5: We generated a new instance for every radius r (as described in the benchmark set), which consists of all demand points that can potentially be covered by some new stop. The effect of the radius on the number of new stops and on the traveling time cannot be clearly predicted since we have two conflicting effects: On the one hand, the number of demand points to be covered increases, we hence may need more stops. On the other hand, every station covers more demand points if r is increased. In our experiments we see that the latter effect is the dominating one. Although there are instances in which the number of stops and the passengers' traveling time increases with increasing radius, the tendency is that increasing the radius leads to fewer new stops and less traveling time.

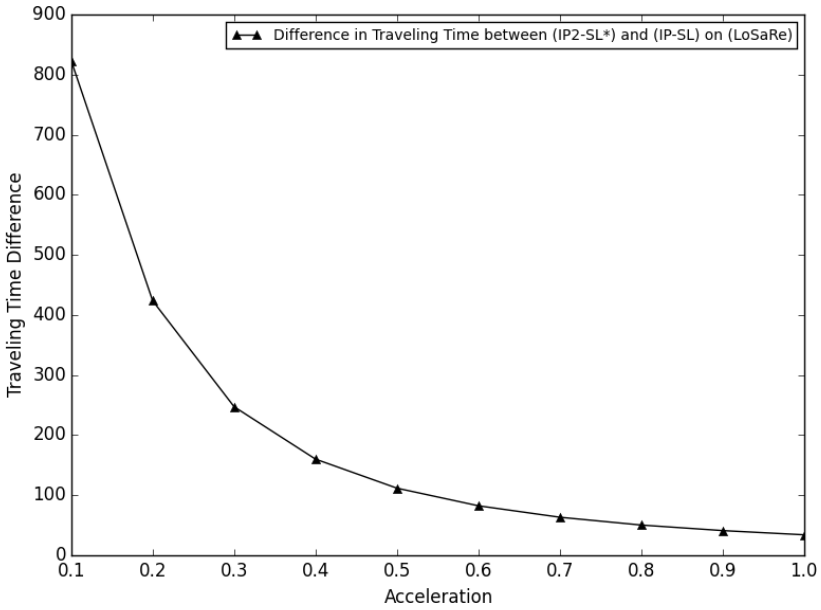


Figure 6: Difference of traveling time values for (IP2-SL*) and (IP-SL) with different acceleration and deceleration values on (LoSaRe)

Dependence on acceleration and deceleration We computed optimal solutions for (SL) and (SL*) on (LoSaRe) for different values for the acceleration and deceleration. For simplicity, acceleration and deceleration are assumed to be equal. As expected, with increasing values of acceleration and deceleration the traveling time values decrease. The traveling time function T behaves in a_0 and b_0 like x^{-1}, which is of a similar shape as the traveling time values depicted in Figure 6. In order to justify Theorem 4.6 we compared the solutions of (SL) and (SL*) in terms of traveling times as follows: Let $a_0 = b_0$ be fixed, then denote by S_{a_0} an optimal solution to (SL) and by $S^*_{a_0}$ an optimal solution to (SL*) for the specific acceleration and deceleration a_0. We evaluate the difference of travel times between these solutions, i.e.,

$$h(a_0) := g(S_{a_0}) - g(S^*_{a_0}).$$

Figure 6 shows this function h. We recognize that the gap between the two traveling time values decreases for increasing values for acceleration and deceleration. It means that the smaller the acceleration and deceleration, the higher the difference between the traveling time values for (SL) and (SL*), i.e., the bigger the error when using (SL) instead of (SL*). Hence, for smaller acceleration and deceleration values such as in train transportation it is

more desirable to use the (SL*) model with an accurate computation of the traveling time.

6.3 Summary

From our numerical experiments, we hence conclude

1. (IP2-SL*) is clearly superior to (IP1-SL*).
2. In the experiments, (SL*) leads to the same (minimal) number of stations as (SL) but improves the additional traveling time.
3. The difference between (SL*) and (SL) is larger for small acceleration and deceleration. I.e. in bus transportation one can use the easier model (SL) and expect a good traveling time while it becomes more important to use (SL*) in train transportation.
4. Increasing the covering radius leads to fewer new stops and less traveling time.

7 Conclusion and further research

In this paper we minimized the passengers' traveling time function in stop location problems. We derived a finite dominating set and two IP formulations and showed the applicability of the model on two different benchmark sets. It turns out that the solutions of (SL*) slightly outperform the solutions of (SL) with similar computation time values and with the same (minimal) number of new stops. Even more we have seen two different IP-formulations for (SL*) and their performance. Judging from the experiments, the second of those (IP2-SL*) has a high potential to even solve complex instances.
Further research on this topic goes into two directions. First, we assumed that all vertices of the existing network are built as stops. However, it may be better to close or move some of these. In order to model this appropriately, an integration with line planning is necessary.
Secondly, the traveling time for the passengers could be even more realistic if OD-pairs are considered. Minimizing an OD-pair based traveling time leads to a different model and thus analysis. It would also allow to take congestion into account: if more passengers are boarding or alighting this leads to larger stopping times of the trains at the stops. Integrating routing of passengers into planning problems has not been done for stop location so far, while first approaches exist for line planning (see, e.g., [Sch14, SS15a]), timetabling

([SS15b, BHK15]), or delay management ([DHSS12, Sch13]). It is a challenging task to develop similar integrated approaches also for stop location.

References

[BHK15] R. Borndörfer, H. Hoppmann, and M. Karbstein. Timetabling and passenger routing in public transport. Technical Report 15–31, ZIB Report, 2015. to appear in Proceedings of CASPT 2015.

[CPR94] J. Current, H. Pirkul, and E. Rolland. Efficient algorithms for solving the shortest covering path problem. *Transportation Science*, 28(4):317–327, 1994.

[DDW09] Z. Drezner, T. Drezner, and G. O. Wesolowsky. Location with acceleration–deceleration distance. *European Journal of Operational Research*, 198(1):157–164, 2009.

[DHSS12] T. Dollevoet, D. Huisman, M. Schmidt, and A. Schöbel. Delay management with rerouting of passengers. *Transportation Science*, 46(1):74–89, 2012.

[Gle75] J. Gleason. A set covering approach to bus stop allocation. *Omega*, 3:605–608, 1975.

[HLS+01] H.W. Hamacher, A. Liebers, A. Schöbel, D. Wagner, and F. Wagner. Locating new stops in a railway network. *Electronic Notes in Theoretical Computer Science*, 50(1), 2001.

[KMP+12] M.-C. Körner, J.A. Mesa, F. Perea, A. Schöbel, and D. Scholz. A maximum trip covering location problem with an alternative mode of transportation on tree networks and segments. *TOP*, 2012. published online, DOI 10.1007/s11750-012-0251-y.

[KPS+03] E. Kranakis, P. Penna, K. Schlude, D.S. Taylor, and P. Widmayer. Improving customer proximity to railway stations. In *CIAC*, pages 264–276, 2003.

[KWH75] P. Kolesar, W. Walker, and J. Hausner. Determining the relation between fire engine travel times and travel distances in new york city. *Operations Research*, 23:614–627, 1975.

[LMO02] G. Laporte, J.A. Mesa, and F.A. Ortega. Locating stations on rapid transit lines. *Computers and Operations Research*, 29:741–759, 2002.

[MDSF98] A. Murray, R. Davis, R.J. Stimson, and L. Ferreira. Public transportation access. *Transportation Research D*, 3(5):319–328, 1998.

[MMW04] M.F. Mammana, S. Mecke, and D. Wagner. The station location problem on two intersecting lines. *Electronic Notes in Theoretical Computer Science*, 92:52–64, 2004.

[MSW06] S. Mecke, A. Schöbel, and D. Wagner. Stop location - complexity and approximation issues. In *5th workshop on algorithmic methods and models for optimization of railways*, number 06901 in Dagstuhl Seminar proceedings, 2006.

[Mur01] A. Murray. Strategic analysis of public transport coverage. *Socio-Economic Planning Sciences*, 35:175–188, 2001.

[Mur03] A. Murray. A coverage models for improving public transit system accessibility and expanding access. *Annals of Operations Research*, 123:143–156, 2003.

[MW03] A. Murray and X. Wu. Accessibility tradeoffs in public transit planning. *J. Geographical Syst.*, 5:93–107, 2003.

[MW04] S. Mecke and D. Wagner. Solving geometric covering problems by data reduction. In *Proceedings of European Symposium on Algorithms (ESA)*, pages 760–771, 2004.

[PHHS09] D. Poetranto, H.W. Hamacher, S. Horn, and A. Schöbel. Stop location design in public transportation networks: Covering and accessibility objectives. *TOP*, 17(2):335–346, 2009.

[RS04] N. Ruf and A. Schöbel. Set covering problems with almost consecutive ones property. *Discrete Optimization*, 1(2):215–228, 2004.

[Sch05] A. Schöbel. Locating stops along bus or railway lines — a bicriteria problem. *Annals of Operations Research*, 136:211–227, 2005.

[Sch06] A. Schöbel. *Optimization in public transportation. Stop location, delay management and tariff planning from a customer-oriented point of view.* Optimization and Its Applications. Springer, New York, 2006.

[Sch13] M. Schmidt. Simultaneous optimization of delay management decisions and passenger routes. *Public Transport*, 5(1):125–147, 2013.

[Sch14] M. Schmidt. *Integrating Routing Decisions in Public Transportation Problems*, volume 89 of *Optimization and Its Applications*. Springer, 2014.

[SHLW09] A. Schöbel, H.W. Hamacher, A. Liebers, and D. Wagner. The continuous stop location problem in public transportation. *Asia-Pacific Journal of Operational Research*, 26(1):13–30, 2009.

[SS15a] M. Schmidt and A. Schöbel. The complexity of integrating routing decisions in public transportation models. *Networks*, 65(3):228–243, 2015.

[SS15b] M. Schmidt and A. Schöbel. Timetabling with passenger routing. *OR Spectrum*, 37:75–97, 2015.

[Vuc81] V. Vuchic. *Urban Public Transportation: Systems and Technology*. Prentice-Hall, Englewood Cliffs, NJ, EU, 1981.

[WM05] C. Wu and A. Murray. Optimizing public transtit quality and system access: the multiple-route, maximal covering/shortest path problem. *Environment and Planning B: Planning and Design*, 32:163–178, 2005.

Addendum B

P. Gattermann, J. Harbering, A. Schöbel
Generation of Line Pools
Proceedings of the 13th Conference on Advanced Systems in Public Transportation: Theory and Applications

Generation of Line Pools

Philine Gattermann (p.gattermann@stud.uni-goettingen.de)
Georg-August-University of Göttingen

Jonas Harbering (jo.harbering@math.uni-goettingen.de)
Georg-August-University of Göttingen

Anita Schöbel (schoebel@math.uni-goettingen.de)
Georg-August-University of Göttingen

December 11, 2015

Finding the lines and their frequencies in public transportation is the well-studied line planning problem. In this problem, it is common to assume that a line pool consisting of a set of potential lines is given. The goal is to choose a set of lines from the line pool that is convenient for the passengers and has low costs. The chosen lines then form the *line plan* to be established by the public transportation company. The line pool hence has a significant impact on the quality of the line plan. The more lines are in the line pool, the more flexible can we choose the resulting line plan and hence increase its quality. It hence would be preferable to allow *all* possible lines to choose from. However, the resulting instances of the line planning problem get too large if all lines would be allowed.

In this work, we study the effect of line pools for line planning models and propose an algorithm to generate 'good' line pools. To this end, we formally introduce the *line pool generation problem* and investigate its properties. The line pool generation problem asks for choosing a subset of paths (the line pool) of limited cardinality such that in a next step a good line concept can be constructed based on this subset. We show that this problem is NP hard. We then discuss how reasonable line pools may be constructed. Our approach allows to construct line pools with different properties and even to engineer the properties of the pools to fit to the objective function of the line planning model to be used later on. Our numerical experiments on close-to real-world data show that the quality of a line plan significantly depends on the underlying line pool, and that it can be influenced by the parameters of our approach.

Keywords: line pool, line planning, public transport, algorithm design

1 Introduction

In public transportation planning the construction of the lines to be used plays a key role for all subsequent planning stages, such as setting the frequencies, finding the timetable and so on. Hence, the scope of line planning should be investigated thoroughly and limitations should be fathomed. Line planning has been extensively investigated in the literature, see [16, 12] for surveys on the topic. The line planning problem is defined as follows:

Given a public transportation network $PTN = (V, E)$ with its set of stops or stations V and its set of direct connections E, a *line* is a path in the PTN. It is common to assume that a *line pool* $\mathcal{L}$ is given consisting of potential lines l, each of them with associated costs $cost_l$ for operating a vehicle on this line. The goal is to choose a *line concept*, i.e., a subset of lines together with their frequencies f_l, such that some given demand is satisfied. There are many possible models for determining a line concept which have been published in the literature. In the so-called *cost models*, the objective function minimizes the costs $\sum_{l\in\mathcal{L}} f_l cost_l$ of the lines chosen (see, e.g., [19, 4]) while in the *direct travelers approach* the goal is to maximize the number of direct travelers (as introduced in [6] and further investigated in [3, 2]). *Travel time models* minimize the approximated traveling time over all OD-pairs (see [17, 15]). In all these models it is assumed that the line pool is given beforehand as input. This may be very restrictive, in particular if the line pool is small.

Instead of choosing lines from a pre-specified line pool, it would hence be preferable to allow all possible paths through the public transportation network as lines. Since a pool consisting of all these possible lines would be by far too large, one idea is to generate the lines within the line planning process. Only few work has been published on this simultaneous generation of lines within the line planning process. [1] developed a model in which lines are generated during the optimization within a column generation approach. However, the subproblem of generating the lines turns out to be a longest path problem which is NP-hard in general, hence the model is computationally hard to solve. The following two early approaches already contain elements of of the line pool generation problem and should hence be mentioned here: In [18] an algorithm for line planning is proposed for which a line pool, called route set in this paper, is computed. This is done by repeatedly adding lines to the already existing route set which improve the existing set of lines from a passenger oriented perspective. [14] iteratively computes different sets of feasible lines which could also be interpreted as line pools. In the first iteration, lines are generated as shortest paths between any pair of leaves of the network. These lines are then adapted by a neighborhood search techniques in subsequent steps.

We also mention that in the transportation (or transit route) network design problem (see [5, 7, 11] for examples and [12, 8] for overviews), it is common to compute the lines in a first step and assign frequencies to them later on. If the frequencies are allowed to be zero, the first

step could also be interpreted as line pool generation. However, the difference is that in the network design problem, usually all lines generated in the first step are used, and hence only few lines are generated. This is far from the intention of our paper in which we generate a large and broad set of lines and allow the existing line planning algorithms in a second step to choose the ones which should really be used.

In this work we present an approach which lies between the two extremes of either having a fixed line pool or of generating the lines within the planning process, namely, we discuss how suitable line pools may be constructed in a pre-step before the actual line planning phase. Our approach allows to generate lines that serve certain purposes. Using our line pools within different line planning models, the influence of the line pool on the resulting optimal solution can be understood better such that we are able to give recommendations on properties lines should have in order to allow a good line concept.

2 Line Planning in Public Transportation

The idea is to split the line planning phase into the following two steps:

Phase 1 Construct a line pool $\mathcal{L}$.

Phase 2 Take $\mathcal{L}$ as input for the line planning model which should be used for finding a line concept.

2.1 Phase 2: Finding a line concept

Before we deal with the generation of a line pool in Phase 1, we look at Phase 2 and describe formally what a line concept is and when it is feasible. Let a public transportation network $PTN = (V, E)$ with nodes V and edges E be given, and assume that we already know a line pool $\mathcal{L}$ of potential lines. A *line concept* assigns a frequency $f_l \in \mathbb{N}_0$ to every line l in the line pool. If $f_l = 0$ then the line l is not used and hence not part of the line plan. If $f_l > 0$ then the line is operated with the frequency f_l, i.e., f_l times in the planning period. As already mentioned in the introduction there are many different models for finding a line concept (see [16] for a survey), common objective functions include minimizing the costs, maximizing the number of direct travelers or minimizing some approximation of the traveling time. Also the constraints differ from model to model. As [16] points out, there is one basic set of constraints which is used in most models for line planning, namely the edge frequency constraints: In order to allow all passengers to travel, it is common to assume a *lower frequency bound* $f_e^{\min}$ on each edge $e \in E$. It specifies the least amount of vehicles that have to traverse each edge. Analogously, also an *upper frequency bound*, denoted by f_e^{max} for all $e \in E$, is often given. This number represents the

maximum number of vehicles that are allowed to traverse each edge and may model capacity or security restrictions. For a line concept with frequencies $f_l \in \mathbb{N}_0$ for all $l \in \mathcal{L}$ it is then required that

$$f_e^{min} \leq \sum_{\substack{l \in \mathcal{L}: \\ e \in l}} f_l \leq f_e^{max} \qquad \forall\, e \in E. \tag{1}$$

We finally repeat three line planning models which are used later on for evaluating our line pools:

(LP0) Basic line planning problem:

Find $f_l \in \mathbb{N}_0$ for all $l \in \mathcal{L}$ such that (1) is satisfied.

(LP-Cost) Cost model: Given cost values $cost_l$ for operating a vehicle on line l for all lines in $\mathcal{L}$, a simple cost model is given as follows.

$$\min \left\{ \sum_{l \in \mathcal{L}} f_l \, cost_l \text{ s.t. (1) is satisfied and } f_l \in \mathbb{N}_0 \text{ for all } l \in \mathcal{L}. \right\}$$

(LP-Direct) Direct passengers model: A direct passenger is a passenger who can travel between his origin and his destination on a preferable path (the preferable paths are given beforehand, e.g. as the shortest paths) without a transfer.

$$\max\{\text{number of direct passengers s.t.} \quad (1) \text{ is satisfied and } f_l \in \mathbb{N}_0 \text{ for all } l \in \mathcal{L}.\}$$

Note that in order to count a passenger as a direct passenger the *capacity* of the vehicles in combination with the frequencies has to be high enough.

2.2 Phase 1: The line pool generation problem

We now turn our attention to Phase 1: We assume that any cycle free path in the PTN is a valid line. Let $\mathcal{L}^0$ be the set of all cycle free paths in the PTN. We then aim to find a set $\mathcal{L} \subseteq \mathcal{L}^0$.

When constructing such a line pool, two conflicting goals need to be considered: On the one hand, the line pool should be large enough to allow the line planning model in Phase 2 to find a good line concept. On the other hand, the larger the line pool is, the more complex is it to solve the line planning model in Phase 2. Hence, we require the number of lines in the line pool to be bounded by a value K to ensure tractability of Phase 2. To ensure that the line pool is 'good

enough' is much harder. A minimal requirement for a good line pool is that it should at least be *feasible* for the line planning problem in Phase 2. Since there are so many different models for line planning it is unrealistic to guarantee feasibility for all possible models. However, the edge frequency constraints (1) are a minimal requirement which is used in most line planning models, and which we hence consider for designing a line pool.
We can now define the *line pool generation problem* for given K as follows.

(**LPool**) Given the PTN=(V, E), lower and upper frequency bounds $f_e^{\min} \leq f_e^{\max}$ for all $e \in E$, and the set $\mathcal{L}^0$ of all cycle free paths in the PTN, find a set $\mathcal{L} \subseteq \mathcal{L}^0$ such that $|\mathcal{L}| \leq K$ and such that there exist $f_l \in \mathbb{N}_0$ for all $l \in \mathcal{L}$ with

$$f_e^{min} \leq \sum_{\substack{l \in \mathcal{L}: \\ e \in l}} f_l \leq f_e^{max} \qquad \forall\, e \in E.$$

Using binary variables

$$x_l = \begin{cases} 1 & \text{if line } l \text{ is chosen to be in the line pool} \\ 0 & \text{otherwise} \end{cases}$$

and variables $f_l \in \mathbb{N}_0$ to ensure the existence of a line concept, LPool can be formulated as the followinig integer program. To this end, let $M \geq \max_{e \in E} f_e^{\max}$.

$$\begin{aligned}
\text{Find} \quad & x_l & & & & (2)\\
\text{s.t.} \quad & \sum_{l \in \mathcal{L}^0} x_l & \leq & K & & (3)\\
& \sum_{\substack{l \in \mathcal{L}^0: \\ e \in l}} f_l & \geq & f_e^{min} & \forall\, e \in E & (4)\\
& \sum_{\substack{l \in \mathcal{L}^0: \\ e \in l}} f_l & \leq & f_e^{max} & \forall\, e \in E & (5)\\
& f_l & \leq & x_l M & \forall l \in \mathcal{L}^0 & (6)\\
& f_l & \in & \mathbb{N}_0 & \forall l \in \mathcal{L}^0 & (7)\\
& x_l & \in & \{0,1\} & \forall l \in \mathcal{L}^0 & (8)
\end{aligned}$$

Note that this is a feasibility problem only. We are mainly interested in the variables x_l since these variables determine the line pool $\mathcal{L} = \{l : x_l = 1\}$. However, a line pool is only feasible if there exists at least one line concept satisfying the basic constraint (1). This is ensured by constraints (4) and (5) and our additional variables f_l. The number of lines in the pool is bounded by constraint (3). (6) finally is needed to ensure that $f_l = 0$ if $x_l = 0$, i.e., if the line

is not in the pool.

In order to compare this formulation with classic line planning models, we look at the special case for $K \geq |\mathcal{L}^0|$. In this case, constraint (3) is redundant. We hence may set $x_l = 1$ for all lines $l \in \mathcal{L}^0$ meaning that also (6) can be dropped from the IP formulation. Consequently, we are only left with the variables f_l, i.e., in the special case of $K \geq |\mathcal{L}^0|$ (2)-(8) is equivalent to

$$\begin{aligned}
\text{Find} \quad & f_l & & & & (9)\\
\text{s.t.} \quad & \sum_{\substack{l \in \mathcal{L}^0: \\ e \in l}} f_l & \geq & f_e^{min} & \forall\, e \in E & (10)\\
& \sum_{\substack{l \in \mathcal{L}^0: \\ e \in l}} f_l & \leq & f_e^{max} & \forall\, e \in E & (11)\\
& f_l & \in & \mathbb{N}_0 & \forall l \in \mathcal{L}^0. & (12)
\end{aligned}$$

which is the basic line planning problem LP0 from above for the line pool $\mathcal{L}^0$. If $\mathcal{L}^0$ is given as an arbitrary set of lines, it is known that this problem is NP-hard, see [4]. However, here $\mathcal{L}^0$ has a special form: it consists of all cycle free paths. This makes the problem (9)-(12) easily solvable: Namely, choose a line

$$l_e := \{e\}$$

for each edge $e \in E$ which consists of the respective edge only, and set

$$f_{l_e} := f_e^{\min} \text{ for all } e \in E.$$

The question now arises if LPool is NP-hard for smaller K even in the case that $\mathcal{L}^0$ contains all cycle free paths. The following theorem answers this question.

Theorem 2.1. *Problem LPool is NP-hard even if $K = 1$.*

Proof. Consider the following reduction of the directed Hamiltonian Path Problem to LPool. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a digraph. Construct an undirected PTN $G = (V, E)$ to determine whether a Hamiltonian Path exists in $\mathcal{G}$. Define

$$\begin{aligned}
V &= \{v_{\text{in}}, v_{\text{out}} : v \in \mathcal{V}\}\\
E &= E_1 \cup E_2\\
E_1 &= \{\{v_{\text{out}}, u_{\text{in}}\} \colon (v, u) \in \mathcal{E}\}\\
E_2 &= \{\{v_{\text{in}}, v_{\text{out}}\} \colon v \in \mathcal{V}\}
\end{aligned}$$

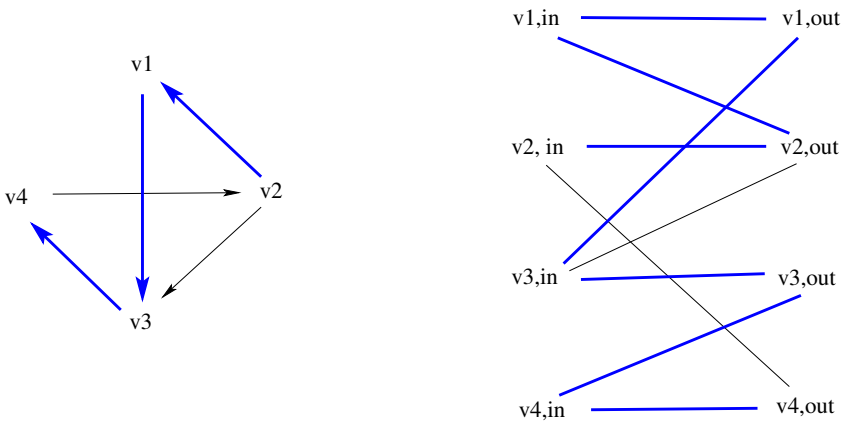


Figure 1: Reduction of Hamiltonian Path to LPool in the proof of Theorem 2.1

The edge set E_1 corresponds to the edges in the original graph while the set E_2 corresponds to the vertices and is used to give the orientation of the edges in $\mathcal{E}$ to the corresponding ones in E_1.
Let $\mathcal{L}^0$ be the set of all possible cycle free paths in G. Set $K = 1$ such that only one line may be chosen for the pool $\mathcal{L}$. Upper frequency bounds are not needed. The lower frequencies $f_e^{\min}$ are set to 0 for all edges in E_1 and to 1 for all edges in E_2. Therefore, all edges connecting in- and outgoing vertices belonging to the same original vertex have to be used once.
Claim: There exists a solution to LPool in G if and only if there exists a Hamiltonian Path in $\mathcal{G}$. (See Figure 1 for an illustration.)

$\Rightarrow$ Let $\mathcal{L}$ be a solution to LPool. Then $\mathcal{L} = \{P\}$ for some path P through the PTN which contains all edges in E_2. In the path P each node has at most degree two and each edge in E_2 is used once. As each node is exactly incident to one edge in E_2, the edges of P are alternately from E_1 and E_2. Since the edges in E_1 go from an outgoing to an incoming vertex, the paths P in G can be transformed into a directed path P' in $\mathcal{G}$ by omitting the edges in E_2. As P uses each edge in E_2, P' traverses each node in $\mathcal{V}$. Therefore, P' is a Hamiltonian Path.

$\Leftarrow$ For a Hamiltonian Path P in $\mathcal{G}$ construct the corresponding undirected path P' in G. Therefore, change the edges from $\mathcal{E}$ into the corresponding ones in E_1 and insert the edges in E_2 for each vertex. This is a cycle free path in G which contains all edges in E_2 as P traverses all vertices $\mathcal{V}$. Using P' as line pool with frequency $f_{P'} = 1$ is, thus, a feasible solution for LPool.

□

Note that the problem is also NP-hard if we allow *all* simple paths in $\mathcal{L}^0$ instead of only cycle free paths.

3 Solution approach for generating a line pool

The algorithm we propose aims at generating line pools which are not too large, allow feasible line concepts and deliver good lines. Our algorithm iteratively adds lines and stops as soon as a feasible line concept is found. The quality of the resulting lines cannot be determined beforehand, but is evaluated experimentally in Section 4.

As input, our algorithm needs a PTN$= (V, E)$. Furthermore, passenger information given as an OD matrix $C = (C_{uv})_{u,v \in V}$ where C_{uv} is the number of passengers who wish to travel from u to v as well as lower and upper frequency bounds $f_e^{min} \leq f_e^{max}$ for all edges E are required. The algorithm is then based on the idea of repeatedly generating paths on minimal spanning trees (MST) of the PTN and adding them to the line pool. The algorithm stops when a feasible line concept is contained in the line pool and hence tends to find line pools which are rather small. Consequently, we do not use a maximum number of lines as input, but determine the size of the line pool afterwards.

3.1 Generation of valid lines

Let a non-empty set of lines $\mathcal{L}$ be given. If this set of lines is not a feasible solution to the line pool generation problem, we have to add further lines to it. A line l to be added to $\mathcal{L}$ has to satisfy constraints on itself and constraints which involve the line l together with the current line pool $\mathcal{L}$. In order to generate lines we proceed as follows:

1. We identify *terminals*.
2. We determine a spanning tree of the PTN with its leaves.
3. We generate lines which connect either two leaves, or two terminals, or a terminal with a leaf.

Let $\mathcal{L} \neq \emptyset$ be a set of lines. For each node we compute its *line degree*

$$deg(v, \mathcal{L}) = |\{l \in \mathcal{L} : v \in l\}|.$$

Let $deg(\mathcal{L})^{\max}$ be the maximum number of lines passing through the same node. Then a node is called a *terminal* if

$$deg(v, \mathcal{L}) \geq \texttt{terminal\_degree} \cdot deg(\mathcal{L})^{\max}$$

for some given parameter `terminal_degree` $\in (0, 1)$. Note that in the very first iteration, we have $\mathcal{L} = \emptyset$. In this case we use the node degree within the PTN instead of the line degree to determine terminals.

In the next step we determine a minimal spanning tree. A *minimal spanning tree* is a connected subgraph G' of the PTN with $G' = (V, E') \subseteq G$, $|E'| = |V| - 1$ and $\sum_{e \in E'} w_e$ minimal. It can be found, e.g., with Kruskal's algorithm ([13]).

With the edge weights for the spanning tree problem we can influence which edges are likely to be contained in the tree. The weight of the edges is changed in each iteration. We denote w_e as the *weight* and l_e as the *length* of edge $e \in E$.

A line l' is called *valid* for a line pool $\mathcal{L}$ and an MST if

- $l' \notin \mathcal{L}$
- it is a terminal-terminal, terminal-leaf or leaf-leaf path
  - if l' is a leaf-leaf path: the euclidean distance between the two end nodes of l' is at least `min_distance`
  - else: l' contains at least `min_edges`.

We end this section by giving a motivation for using leaf-terminal lines: Looking at subgraphs of a tree consisting of a terminal-leaf connection yields a linear graph. In our heuristic, the edges of this linear graph form a single line, with no other line starting or ending in between. In the following proposition we give a reason why such a line should be added to the line pool (if it is valid). This reason is based on the objective function used for finding the line concept in Phase 2: Namely, such a line is likely to be included in a line concept since it has low costs (compared to its length) and is also one with the maximum number of direct travelers.

Proposition 3.1. Let the PTN be a linear graph $G = (V, E)$, $f_e^{min} > 0 \ \forall e \in E$, $K = \infty$, $\mathcal{L}$ the set of all paths in G and the line-costs are of the form $cost_l = F + \sum_{e \in E} c_e$. Here F is a fixed cost-term and c_e are edge-dependent costs.

Let l be the line between the two leaves of the linear PTN, i.e., the line consisting of all edges. Then there

1. exists an optimal solution to LP-Cost which contains line l with frequency $f_l = \min_{e \in E} f_e^{min}$
2. exists an optimal solution to LP-Direct which contains line l if the upper edge frequencies f_e^{max} are large enough.

Proof. 1. Consider an optimal solution to LP-Cost with $f_l < \min_{e \in E} f_e^{min}$. Then there exists a set of lines in $\mathcal{L} \setminus \{l\}$ with positive frequencies, such that each edge is covered at least once. Let $\mathcal{L}' = \{l_1, \ldots, l_m\}$ be such a set, where additionally no line is contained in

another one and the lines are ordered according to their first edge. Now consider the set of lines which are intersections of two consecutive lines

$$\mathcal{L}'' = \{l_1 \cap l_2, l_2 \cap l_3, \ldots, l_{m-1} \cap l_m\} \cup \{l\}.$$

It is easy to see that each edge is covered by $\mathcal{L}'$ as often as by $\mathcal{L}''$. Increasing the frequencies of each line in $\mathcal{L}''$ and decreasing the frequencies of each line in $\mathcal{L}'$ leads to a new solution which can be shown to be feasible and optimal by some calculations and yields a larger value for f_l. Iterating this process until $f_l = \min_{e \in E} f_e^{min}$ proves the claim.

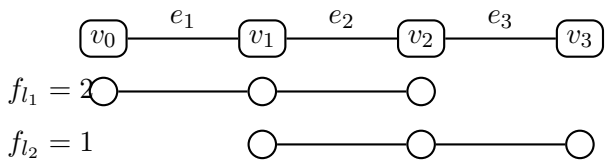
2. The second assertion is clear since every OD-pair in the linear graph can travel without a transfer if line l is contained within the line concept.

□

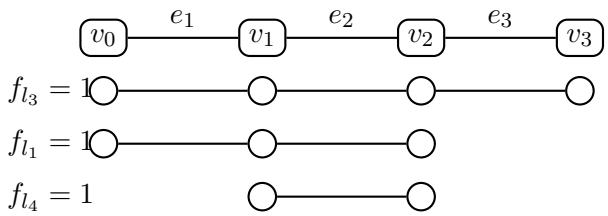
Remark 3.2. We can apply the idea of the proof of Proposition 3.1 to any set of lines covering the same connected component of a linear PTN. Using this approach recursively gives us an optimal solution to LP-Cost in which for all pairs of lines l_1, l_2 with positive frequencies we have that either $l_1 \subseteq l_2$ or $l_2 \subseteq l_1$ or $l_1 \cap l_2 = \emptyset$. For details, see [9].

The following example shows the limitation of adding the longest path as a line when the direct traveler model is concerned. Here, upper frequency bounds and relatively small vehicle capacities may lead to shorter lines being preferred.

Example 3.3. Consider a linear PTN consisting of four nodes $V = \{v_0, \ldots, v_3\}$ and three edges $E = \{e_1, e_2, e_3\}$. Let the upper and lower frequencies for each edge be identical with $f_{e_1}^{\min} = f_{e_1}^{\max} = 2$, $f_{e_2}^{\min} = f_{e_2}^{\max} = 3$ and $f_{e_3}^{\min} = f_{e_3}^{\max} = 1$. Let the capacity of the vehicle be one and the demand $C_{u,v}$ between stations u and v be given as $C_{v_0,v_2} = 2$, $C_{v_1,v_3} = 1$ and $C_{u,v} = 0$ otherwise. Consider the lines $l_1 = (e_1, e_2)$ and $l_2 = (e_2, e_3)$ with the corresponding line concept $f_{l_1} = 2$, $f_{l_2} = 1$ and $f_l = 0$ for all other possible lines l. This line concept allows all passengers to travel directly, so the total number of direct travelers is three.



If the longest line $l_3 = (e_1, e_2, e_3)$ is used with frequency $\bar{f}_{l_3} = 1$, the best possible line concept is $\bar{f}_{l_3} = 1$, $\bar{f}_{l_1} = 1$ and $\bar{f}_{l_4} = 1$ with $l_4 = (e_2)$. Due to the capacity constraint this line concept allows only two direct travelers and is therefore not optimal.



3.2 Generation of a line pool

We finally can describe our algorithm for generating a line pool. The generation of the single lines to be added is based on the theoretical results of the previous section.

The algorithm consists of an outer loop which checks if the line pool found so far is feasible, and if not, determines the parameters for the inner loop, in which spanning trees are constructed and used to generate lines.

In the outer loop (see Algorithm 1) we start with routing all passengers through the network in order to see which edges are used by many passengers. The `pass_edge_ratio`% most used edges are set to be preferred, i.e., their weight for determining an MST is set to zero. All other edges keep their physical length as weight. We then determine a first MST from which we generate valid lines in the inner loop.

All other iterations of the outer loop start with a non-empty line pool. In each iteration we check if this line pool is feasible according to (1). If it is, the algorithm stops. Otherwise we want to identify edges which should be used for generating new lines. To this end, we decrease the lower frequencies until we find a feasible line concept within our pool (lines 4-19). This line concept is not feasible for at least one of the original lower edge frequency constraints. We determine all edges e for which (1) is not satisfied (line 15) and add them to the preferred edges with zero weight in order to make it likely that they are included in the next spanning tree (line 22) In each step of the *inner loop* (25-31) we determine a spanning tree and identify terminal nodes.

We then use the MST and the terminals to generate valid lines in Algorithm 2. This is done by adding additional edges to non-valid lines until they reach the minimal length `min_edges` required for paths starting or ending at a terminal. A valid line is only added to the line pool if it contains at least one preferrable edge, and if it does not increase the frequency of its edges too much, i.e., if

$$|\{l \in \mathcal{L} : e \in l\}| \leq f_e^{max} \cdot \texttt{max\_num\_cover}$$

holds. If a line is added which contains a preferred edge e we might want to remove edge e from the list of preferred edges. This is done if

$$\lceil f_e^{min} / \texttt{min\_num\_cover} \rceil \leq |\{l \in \mathcal{L} : e \in l\}|$$

Algorithm 1 Main function for generating a line pool

```
1: function MAIN
2:     pool= ∅
3:     preferred_edges= ∅
4:     while pool infeasible and number_of_iterations not reached do
5:         if first loop then
6:             Calculate shortest paths for all passengers in PTN.
7:             For each edge determine the number of passengers using it.
8:             Add the pass_edge_ratio most used edges to preferred_edges.
9:         else
10:            if feasible line concept (ℒ, f) found then
11:                return Line pool pool
12:            else
13:                Set f̄^min = f^min
14:                while no feasible line concept found do
15:                    Find a line concept (ℒ̄, f̄) with f̄^min in (1).
16:                    if no such line concept exists then
17:                        set f̄_e^min = f̄_e^min − 1 for all e ∈ E.
18:                    end if
19:                end while
20:            end if
21:            for edge e ∈ E with Σ_{l∈pool:e∈l} f̄_l < f_e^min do
22:                preferred_edges=preferred_edges ∪{e}
23:            end for
24:        end if
25:        while pool increased and preferred_edges ≠ ∅ do
26:            Find MST in PTN, with
27:            w_e = 0 for all e ∈ preferred_edges and
28:            w_e = l_e for all e ∉ preferred_edges.
29:            Determine leaves and terminals based on terminal_degree.
30:            POOL_FROM_MST(pool, MST, terminals, preferred_edges)
31:        end while
32:    end while
33: end function
```

Algorithm 2 Function computing lines based on one MST

```
1: function POOL_FROM_MST(pool, MST, terminals, preferred_edges)
2:     Initialize unfinished_lines = {terminals} ∪ {leafs}
3:     for line ∈ unfinished_lines do
4:         Delete line from unfinished_lines
5:         stop ← last stop in line
6:         for edge ∉ line incident to stop do
7:             Create new_line l' by adding edge to line.
8:             if new_line l' is not valid then
9:                 Add new_line l' to unfinished_lines.
10:            else if |{l ∈ L ∪ {l'} : e ∈ l}| ≤ f_e^max · max_num_cover and
11:                    new_line l' contains an edge from preferred_edges then
12:                Add new_line l' to pool.
13:                for edge e ∈ new_line l' do
14:                    if ⌈f_e^min / max_num_cover⌉ ≤ |{l ∈ pool : e ∈ l}| then
15:                        Remove e from preferred_edges
16:                    end if
17:                end for
18:            end if
19:        end for
20:    end for
21: end function
```

is fulfilled. After a spanning tree is processed, a new spanning tree is constructed and the process is repeated until there are no preferred edges left, or no further line can be added.

Afterwards the feasibility of the pool is checked by solving a line planning problem using the pool. The algorithm returns an error if no feasible line concept can be constructed. Note that the pool may only be infeasible if the maximal number of iterations of the outer loop is reached.

3.3 Summary of Algorithm parameters

We summarize the list of parameters which control the algorithm.

- `pass_edge_ratio`: Used in the first iteration to obtain preferred edges. More precisely, the `pass_edge_ratio` percent of most used edges are chosen as preferred edges.
- `min_num_cover`: For each edge, the lower frequency bound divided by `min_num_cover` bounds the number of lines containing this edge, i.e., for all $e \in E$ we require $\lceil f_e^{min} / \texttt{min\_num\_cover} \rceil \leq |\{l \in \mathcal{L} : e \in l\}|$.
- `terminal_degree`: Deviation from maximum line degree such that a node is labeled terminal.
- `min_edges`: Minimum number of edges per line (only for lines from or to a terminal)
- `min_distance`: Minimum euclidean distance between start and end station (only for lines from and to a leaf)
- `max_num_cover`: For each edge, the upper frequency bound times `max_num_cover` bounds the maximal coverage of this edge, i.e., for all $e \in E$ we require $|\{l \in \mathcal{L} : e \in l\}| \leq f_e^{max} \cdot \texttt{max\_num\_cover}$.

Whereas the first two parameters are only used to determine the edges which are preferred in the computation of the MSTs the other parameters are related to the validity of a line.

3.4 Determining the costs

In order to use the line pool generated by our approach for line planning in Phase 2, we have to add costs for each line in the pool. The costs of a line are influenced by various parameters from which we take the following three into account:

- length of line (`costs_length`)
- number of edges traversed (`costs_edges`)
- fixed costs (`costs_fixed`)

Then the costs for a line $l \in \mathcal{L}$ are computed as

$$cost_l = \texttt{costs\_length} \sum_{e \in l} l_e + \texttt{costs\_edges} \, |\{e \in l\}| + \texttt{costs\_fixed}$$

4 Results

In order to study the performance of Algorithm 1 we apply it to generate line pools in a first step. In the second step we used the cost model and the direct travelers model on the generated line pools.

We used two different network types. The first are star shaped networks. In these graphs we can use an algorithm of [9] to optimally solve the cost model even if all possible paths are allowed as lines. This gives the best possible objective function value which can be used as lower bound for our two-step approach. Our second network is close to the network of the German railways. It is used to show its general applicability for realistically sized and shaped instances.

4.1 Results for stars

Our first data instance set comprises different star shaped graphs.

Data set

We generated 100 star graphs as follows. For every $n \in \{10, 100, 250, 500\}$ we generated 25 stars with a number of edges (and hence nodes) chosen randomly from $\{1, \ldots, n\}$. The upper frequency bound is set to $f_e^{max} = 20$ for all edges and the lower frequency bound is randomly determined between 0 and 20. The costs of the edges are aslo randomly chosen with values between 0 and 10 and the fixed costs of a line are set to 10.

Experiments

The parameter setting for the star shaped graphs is according to Table 1. Note that the parameters `min_num_cover` and `max_num_cover` take two different values dependent on the respective instance. The test instance with at most 500 edges per graph requires too much memory to be computable with the setting used for smaller instances. Hence, we set `min_num_cover` to 4 and `max_num_cover` to 1 here. For all other instances `min_num_cover` and `max_num_cover` are set to 1 and 5, respectively, to allow many lines in the pool. The parameter `min_edges` is set to 1 to allow lines consisting of only one edge and the parameter `min_distance_leaves` is set to 0 to allow lines consisting of any pair of edges. In stars, the parameters `ratio_od` and

`node_degree_ratio` do not have any effect on the algorithm as there is only one possible minimal spanning tree and only one possible terminal. Therefore, these parameters are not changed during the tests.

parameter name	parameter value
`min_num_cover`	1 (4)
`max_num_cover`	5 (1)
`number_of_iterations`	5
`pass_edge_ratio`	0.5
`terminal_degree`	0.5
`min_edges`	1
`min_distance`	0

Table 1: Parameters for Algorithm 1 on randomly generated stars

Evaluation parameters

For every star shaped graph, we can easily compute the optimal objective value of LP-Cost for a line pool which contains all possible simple paths (see [9]). This value is then compared to the objective value of LP-Cost for the line pool obtained from applying Algorithm 1 to the graph.

Results

The results of testing Algorithm 1 on the randomly generated stars are shown in Table 2.

max. number of edges	**10**	**100**	**250**	**500**
av. number of edges	4.72	39.68	139.36	250.96
number of optimal solutions	19	20	4	0
ratio of optimal solutions	0.76	0.80	0.16	0
av. deviation from optimal value	1.03	1.01	1.14	1.36
max. deviation from optimal value	1.21	1.15	1.27	1.41
av. number of undirected lines	12.60	308.52	1242.84	697.28
av. number of possible undirected lines	17.64	1177.40	11957.32	39580.60
av. runtime in seconds	5.36	34.08	97.28	90.64

Table 2: Results for randomly generated stars

We observe that especially for smaller stars (up to 100 edges) the heuristic solution found by Algorithm 1 is very good. In more than 75% of all cases the optimal solution is found by the heuristic and the objective values of the heuristic solutions deviate from the optimal objective value on average only between 1% and 3% and at most by 21%. When using larger stars (up to 250 edges) this quotient increases but the objective values still lie on average only 14% and at

most 27% above the optimal values. That the quality of the line pool gets worse when proceeding to stars with up to 500 edges is mainly due to changing the parameters `min_cover_factor` and `max_cover_factor` which leads to considerably fewer lines in the pools and hence to line pools with reduced quality: On average we obtain 697.28 lines for stars consisting of 250.96 edges opposed to 1242.84 lines for stars consisting of 139.36 edges. But in turn, the decreased number of lines also leads to a decrease in runtime.
Overall, the heuristic Algorithm 1 works quite well especially for smaller stars, although this depends on choosing the parameters such that enough lines are added to the pool. We remark that in order to achieve a good approximation of the optimal solution, only a fraction of all lines is needed in the pool. The discrepancy between the possible number of lines and the number of lines which make up the pools shows that Algorithm 1 is a good example for line pool generation as a preprocessing step for line planning. Especially for larger PTNs the number of lines which have to be considered when finding a line concept is reduced such that further computations are achievable in a reasonable time.

4.2 Results for the German railway network

Data set

The other data set is generated from the German high speed railway network. It consists of $|V| = 250$ nodes and $|E| = 326$ edges. The frequency bounds are given as $f_e^{min} \in [0, 5]$ and $f_e^{max} = 15$ for all $e \in E$. Different settings for each of the seven parameters listed in Section 3.3 are tested.

Experiments

Experiments on the Germain railway network are done with LinTim (see [10]). Based on the data set and one specific parameter setting a line pool is generated. The resulting line pool is then taken as input for the cost line planning model LP-Cost and the direct travelers line planning model LP-Direct, see Section 2.1. These IP programs are solved with FICO Xpress. We obtain the objective function value for both models: the minimal possible costs and the maximal possible number of direct travelers.

The algorithm has been executed with a variety of different parameter patterns. In total 624 different settings have been tested, and for each of them a line pool has been generated.

Evaluation Parameters

The quality of a line pool is evaluated by two rather opposing parameters. The first parameter gives the lowest possible costs which are determined as the objective function value of the cost-

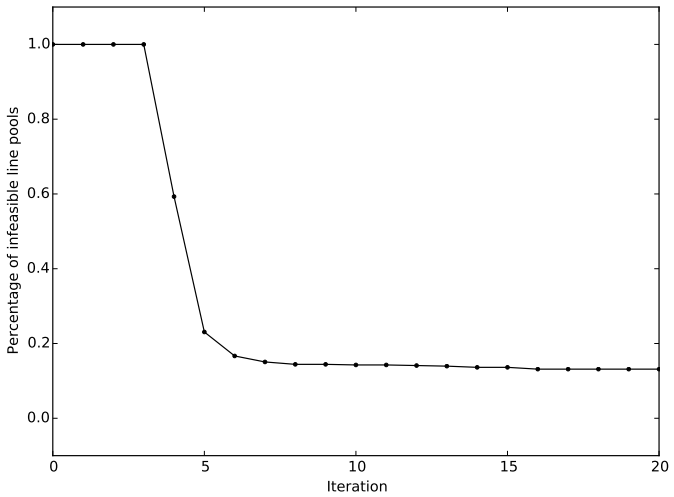


Figure 2: Feasibility of all line pools

optimal line concept. The second parameter reflects the passengers' point of view. Here, we measure the number of direct travelers that can travel on their shortest paths in the PTN in the direct-travelers optimal solution. Let a line pool $\mathcal{L}$ be given.

- Costs: We solve LP-Cost using the costs $cost_l$ for each line $l \in \mathcal{L}$ according to Section 3.4.
- Direct Travelers: We solve LP-Direct. To this end, preferable paths are needed. In our computation we use for each OD-pair $(u, v) \in C$ only one preferable path, namely the shortest path p from u to v in the PTN. The union $\mathcal{P} = \bigcup p$ gives the set of paths for all OD-pairs. Based on the paths $\mathcal{P}$, the number of direct travelers on each $p \in \mathcal{P}$ is computed. For a more detailed description of the definition and optimization of direct travelers we refer to [3].

Results

Feasibility. The first result shows the reliability of the algorithm on the German railway network. Most of the line pools (87%) obtained by any of the parameter settings are feasible solutions to LPool, i.e., they contain a feasible line concept.

We further investigated which particular parameter settings have a recognizable influence on the feasibility.

In Figure 2 the percentage of infeasible line pools in relation to the iteration of Algorithm 1 is depicted. Until iteration 3 no line pool is feasible whereas from iteration 7 on the percentage of feasible line pools only increases slightly. Hence, for most parameter settings between 4 and 7 iterations are needed to receive a feasible line pool.

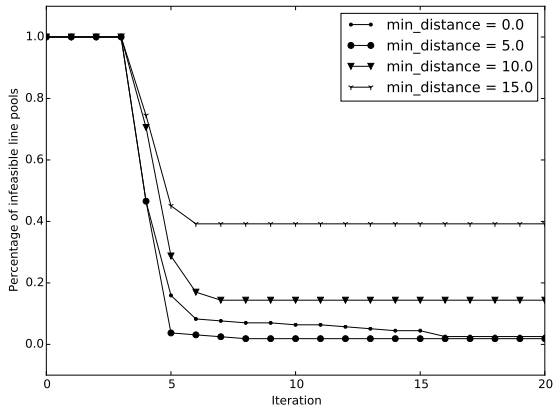


Figure 3: Feasibility of line pools under parameter `min_distance`

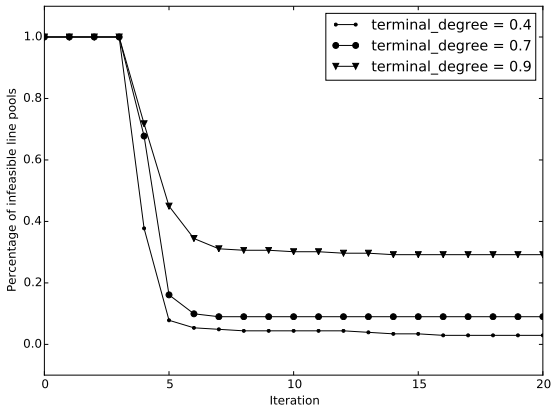


Figure 4: Feasibility of line pools under parameter `terminal_degree`

In Figures 3 and 4 the influence of the parameters `min_distance` and `terminal_degree` on the feasibility of the line pools is studied. It can be recognized that the percentage of feasible line pools is influenced by the choice of these parameters. E.g. requiring that the end nodes of leaf-leaf lines are far from each other, i.e. `min_distance`= 15, results in 39% of infeasible line pools after 6 iterations. Setting `min_distance` to a smaller value, e.g. 5, only 2% of infeasible line pools are obtained after 7 iterations.

The percentage of feasible line pools also depends on the setting of `terminal_degree`. While a value of 0.9 results in 31% of infeasible line pools after 7 iterations, setting the parameter to 0.7 results in only 10% of infeasible line pools.

Considering all tested settings, minus the ones with `min_distance`= 15 and `terminal_degree`= 0.9, the percentage of feasible line pools after 7 iterations reaches 95% and even 97% after 16 iterations. The other parameter settings do not show any significant effects on the feasibility of the resulting pools.

Running times. In addition to the feasibility, also the running times of the subsequent line planning models are important. The computation times for subsequent line planning models are low as the number of lines in the pools range between 115 and 342. The range of computation time for the cost line planning model is from 1.0 to 49.0 seconds, with a mean of 1.59 seconds. For the direct travelers model, the computation time ranges between 6.0 and 94.0 seconds and is on average 11.95 seconds.

What are the influences of the parameters? We now want to study the impact of different algorithm parameters on the evaluation parameters. As described, the resulting line pools are used as input for the line planning problem using the cost model and the direct travelers

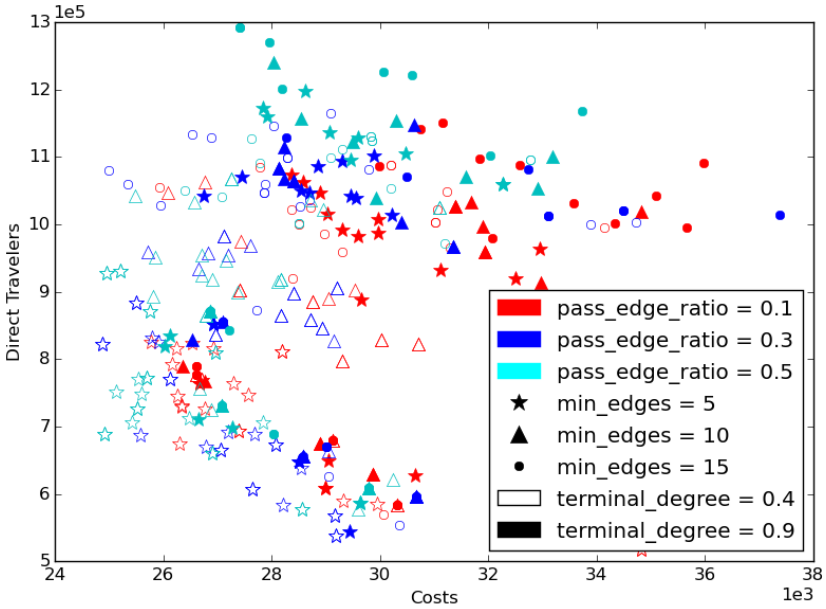


Figure 5: Different line pools evaluated for costs and direct travelers

approach. The result is indicated in Figure 5. Each point in this figure reflects a line pool. It shows the objective values of the cost model and of the direct travelers model. In order to understand the influence of the parameters we additionally show three attributes for each point which represent the parameters used for generating the respective line pool. A point is specified by a particular setting for `min_edges` (star, triangle, circle), `pass_edge_ratio` (red, blue, cyan) and `terminal_degree` (contour, filled).

How long should a line be? We start with an analysis of the parameter `min_edges` which specifies how many edges are required in a line.

Firstly, in order to show the implication of the choice of `min_edges`, we study how many edges the lines in the line pools have. For every pool we evaluate the number of edges in the shortest line and the average number of edges per line in the pool. The question is how these values are influenced by the parameter `min_edges`.

Setting `min_edges` to 5 results in an average number of 10.09 edges per line and the shortest line having 2 edge. Increasing `min_edges` to 10 the average number of edges per line also increases to 11.82 whereas the shortest line still has 2 edges. Finally, setting `min_edges` to 15 gives the same number of edges in the shortest line and the average number of edges in a line again raises to 13.89. The same structure can indeed be found in the resulting line concepts.

While the shortest line always contains 2 edges, the average number of edges per line increases

if min_edges is increased. Analyzing these shortest lines, we note that only leaf-leaf lines are not restricted in the number of edges. Hence, lines shorter than min_edges are only possible if leaf-leaf lines exist whose endpoints have a larger euclidean distance than min_distance. If the parameter min_distance is set to a larger number, then increasing min_edges in fact also leads to more edges in the shortest line. This is confirmed by our experiments: Choosing min_distance as 5 we obtain that the shortest line contains 5 edges for any setting of min_edges. Increasing min_distance to 15 the shortest line contains exactly as many edges as specified by min_edges.

We summarize that there is a clear effect of the parameter min_edges on the average number of edges per line. The minimal number of edges per line is then influenced by both parameters min_edges and min_distance.

We now study the implication of the parameter min_edges to our evaluation parameters. A clear correspondence to the number of direct travelers can be observed. A value of 5 for min_edges results in an average number of direct travelers of 838761.65. The values 10 and 15 result in average direct travelers of 901603.33 and 958793.89 respectively. It indicates that the more edges are required per line the higher the number of direct travelers until some upper bound is reached. In our computations it turned out that setting min_edges to 20 gives no significant improvement in relation to min_edges=15. A relation between the parameter min_edges and the costs of the cost optimal line concept cannot be recognized.

How many terminals should be generated? We now turn our attention to the parameter terminal_degree which determines how many terminals are generated in each step. Figure 5 shows that generating less terminals produces better line pools in terms of the direct travelers. The relation is understandable as lines have to combine either leaf and leaf, terminal and leaf or terminal and terminal nodes. Reducing the number of terminals has two effects: First, fewer terminals lead to longer lines and longer lines are more likely to be direct connections for passengers than shorter lines. Moreover, the less terminals are available, the more leaf to leaf lines are obtained, which allow more passengers to travel directly instead of transferring at terminal nodes. Indeed, choosing terminal_degree as 0.4 the average number of direct travelers results in 850925.84. Restricting the number of terminals by using a value of 0.7 for terminal_degree, an average number of 928457.54 direct travelers is obtained. Increasing the value for terminal_degree further does not seem to make an impact as the average number of direct travelers for terminal_degree= 0.9 is 927327.91.

Figure 5 does only show a slight correlation between the parameter terminal_degree and the costs of the cost optimal line concept. Choosing terminal_degree as 0.4 leads to average costs of 28109.94, while average costs of 29011.20 are obtained by setting this parameter to 0.7. Setting the paramter to 0.9 finally results in an average cost of 29804.12. Hence, small values

of `terminal_degree` seem to have the tendency of reducing the costs of the cost optimal line concept.

How many lines should pass through an edge? The parameters `min_num_cover` and `max_num_cover` have different effects. The parameter `min_num_cover` ensures that each edge is traversed by enough lines. The higher this parameter, the fewer lines are needed to cover each edge. Hence, with increasing value of `min_num_cover` the average number of lines per pool decreases. This seems to have a negative effect on the average number of direct travelers as the following numbers show: For `min_num_cover`= 1, the average number of lines in the pool is 481.49 and the average number of direct travelers is 950488.90 while for `min_num_cover`= 2 (or 4) the line pool size is on average 358.60 (306.12) and the number of direct travelers is on average 887568.32 (857291.88). An effect on the costs can also be recognized. `min_num_cover`= 1 results in average costs of 27698.52. Increasing this value to 2 (and 4), also the costs increase to 29273.31 (and 29574.48). This may be due to the fact that the lines get shorter and hence more expensive compared to their lengths.

The parameter `max_num_cover` specifies the maximum number of lines per edge. Our experiments show that this parameter does not influence the solution quality. It is also recognizable that lines under the setting `max_num_cover`= 3 have slightly less edges than for `max_num_cover`= 5. The difference between the averages is about 0.8 edges but this difference does not show an impact on the solution quality.

How far should start- and endpoint of a line be apart from each other? We already saw some effects of the parameter `min_distance` (specifying the minimal euclidean distance between the start - and the endpoint of a line) together with the minimal number of edges of a line. Further results concerning this parameter are depicted in Figure 6. In particular, we tested the outcome if this condition is completely ommitted, i.e., if also lines are allowed that connect two endpoints that are close to each other (and hence may make very large detours). To this end we compared the solutions for `min_distance`= 0 with `min_distance`= 15.

The figure shows that allowing lines between any pair of leaf nodes (i.e., for the case of `min_distance`= 0) has no effect on the costs (except of some outliers with high cost values in the case of `min_distance`= 15) but decreases the best possible number of direct travelers. The average number of direct travelers for `min_distance`= 0 is 694184.77 while the average over all experiments with `min_distance`> 0 was 985411.94.

Other observations. Finally, The parameter `pass_edge_ratio` does not seem to have a strong impact on any evaluation parameter. This can be explained, since the parameter is only

important in the first outer loop, where according to this parameter passenger weighted edges are preferrably considered in the minimal spanning tree.

We also investigated if there is a correspondence between the number of lines in the line pools we generated and the quality of the evaluation parameters. It turned out that the costs get only a little bit better for larger line pools while there is no clear relation to the number of direct travelers.

Summary. We conclude that for having good line concepts, it is not so important *how many* lines are chosen, but it is crucial *which* lines we take. This underlines the fact that it is possible to generate good line pools of moderate size.

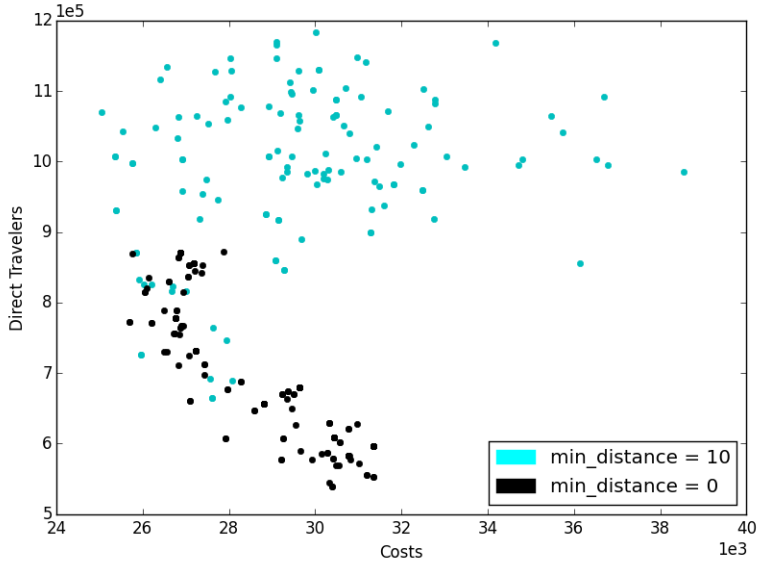


Figure 6: Different line pools evaluated for costs and direct travelers, color coding by `min_distance` parameter.

We summarize our findings as follows:

- Increasing `min_edges`, i.e., requiring a high minimum number of edges per line improves the number of direct travelers (up to some bound).
- Increasing `min_num_cover`, i.e., the number of lines that have to traverse an edge, reduces the number of direct travelers and increases the costs.
- Increasing `min_distance` (i.e., allowing only lines between stations with a minimal euclidean distance) leads to more direct travelers.

- Increasing `terminal_degree` (i.e., allowing less terminals) increases the number of direct travelers and has a weak tendency of decreasing the costs.
- The values of `max_num_cover` and `pass_edge_ratio` influence the line pool, but there is no clear correlation to the objective function values.
- Line pools with many lines do not necessarily allow better line concepts. The specific choice of lines (determined by the algorithm parameters) is more important.

5 Extension of the algorithm: Adding shortest paths to the line pool

In the literature, lines are sometimes constructed in a rather naive approach of just taking all pairwise shortest paths for all $u, v \in V$ and removing all lines which are contained in other lines. In our setting this returned a line pool of more than 2000 lines. Such a line pool size is far more than what is desired and what can be properly handled by OD-pair related line planning problems such as LP-Direct.

However, we allow to add shortest paths to our line pool. This is done by using Algorithm 3 after Algorithm 1 terminated.

Algorithm 3 Function for adding shortest paths as lines to the line pool

```
1: function ADD_SHORTEST_PATHS(pool)
2:     if shortest paths are to be added as lines then
3:         for u, v ∈ V do
4:             if C_{u,v} ≥ max_{u',v'} C_{u',v'} · ratio_sp then
5:                 Add shortest path from u to v to pool.
6:             end if
7:         end for
8:     end if
9: end function
```

In order to control which and how many shortest paths are chosen as lines the parameter `sp_ratio` is introduced. If the demand between two stations $u, v \in V$ is at least `sp_ratio` of the maximal demand between any two stations, the shortest path from u to v is added as a line to the pool.

The feasibility of a pool is tested after the shortest paths are added as lines in the same way as the feasibility of a pool constructed by Algorithm 1.

We also evaluated this step experimentally. The results are depicted in Figure 7. Changing from no additional shortest paths in the line pool to `sp_ratio`$= 0.5, 0.7$ slightly increases the

number of lines in the pool from 190.52 to 197.01 and 209, 33, respectively. On the other hand, some benefits both in terms of the costs and direct travelers are gained. The costs are decreased from 28863.92 to 28329.74 and 27789.67, respectively. The direct travelers, in turn, are increased from 897935.13 to 902885.91 and 917859.82, respectively. Hence, adding some shortest paths increases the line pool and also has a small, but visible positive effect on both the costs and the number of direct travelers.

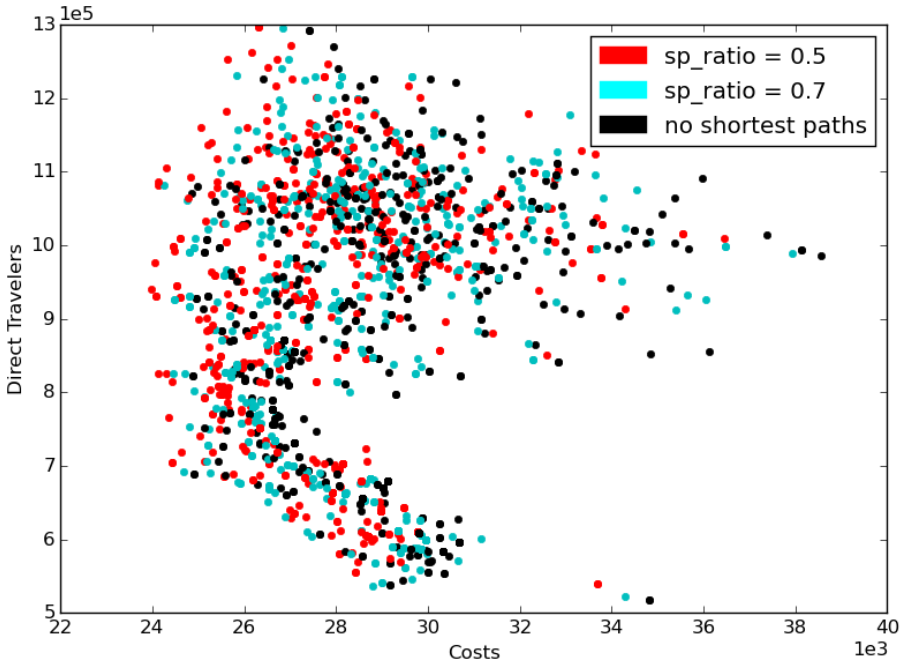


Figure 7: Different line pools evaluated for costs and direct travelers, color coding by `sp_ratio` parameter.

6 Outlook

In this work we study the influence of the line pool to line planning. We propose an algorithm taking care of the line pool construction in a reliable manner. In our tests, most of the generated line pools are feasible and guarantee low running times for the subsequent line planning algorithms.

We have seen that choosing different parameters for the algorithm results in very different line pools, and we sketched the resulting correlations. More systematic and detailed numerical tests with an additional statistical analysis could give a clearer picture of the actual relations

between the algorithm parameters and the evaluation parameters. We also work on line pool generation on simple graphs such as linear or star-shaped networks which will yield a better understanding of the general problem.

The problem of designing a line pool can also be seen as a line planning problem under uncertainty: At the stage of designing the pool, the network PTN is more or less known, but the cost parameters and the lower and upper frequency requirements are usually not known. While there is a need to design lines, the frequencies may be adapted in a second stage. Such a two-stage approach would mean to fix only the line pool (i.e., the x variables in the IP formulation of LPool) and deal with the frequencies after the real paramters are known. This interpretation as adjustable robust problem is currently under research.

Acknowledgement

The authors were partially funded by by the European Union Seventh Framework Programme (FP7-PEOPLE-2009-IRSES) under grant number 246647 with the New Zealand Government (project OptALI). We also thank the Simulationswissenschaftliches Zentrum Clausthal-Göttingen (SWZ) and the DFG funded Research Unit FOR2083 for financial support.

References

[1] R. Borndörfer, M. Grötschel, and M.E. Pfetsch. A column generation approach to line planning in public transport. *Transportation Science*, 41:123–132, 2007.

[2] M.R. Bussieck. *Optimal lines in public transport.* PhD thesis, Technische Universität Braunschweig, 1998.

[3] M.R. Bussieck, P. Kreuzer, and U.T. Zimmermann. Optimal lines for railway systems. *European Journal of Operational Research*, 96(1):54–63, 1996.

[4] M.T. Claessens, N.M. van Dijk, and P.J. Zwaneveld. Cost optimal allocation of rail passenger lines. *European Journal on Operational Research*, 110:474–489, 1998.

[5] K.M. Curtin and S. Biba. The transit route arc-node service maximization problem. *European Journal of Operational Research*, 208(1):46–56, 2011.

[6] H. Dienst. *Linienplanung im spurgeführten Personenverkehr mit Hilfe eines heuristischen Verfahrens.* PhD thesis, Technische Universität Braunschweig, 1978. in German.

[7] L. Fan and C.L. Mumford. A metaheuristic approach to the urban transit routing problem. *Journal of Heuristics*, 16(3):353–372, 2010.

[8] R.Z. Farahani, E. Miandoabchi, W.Y. Szeto, and H. Rashidi. A review of urban transportation network design problems. *European Journal of Operational Research*, 229(2):281–302, 2013.

[9] P. Gattermann. Generating line-pools. Master's thesis, Universität Göttingen, 2015.

[10] M. Goerigk, J. Harbering, and A. Schöbel. LinTim - Integrated Optimization in Public Transportation. Homepage. see http://lintim.math.uni-goettingen.de/, 2015.

[11] J.F. Guan, H. Yang, and S.C. Wirasinghe. Simultaneous optimization of transit line configuration and passenger line assignment. *Transportation Research Part B: Methodological*, 40(10):885–902, 2006.

[12] K. Kepaptsoglou and M. Karlaftis. Transit route network design problem: Review. *Journal of transportation engineering*, 135(8):491–505, 2009.

[13] J.B. Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical society*, 7(1):48–50, 1956.

[14] C. Mandl. Applied network optimization. 1979.

[15] K. Nachtigall and K. Jerosch. Simultaneous network line planning and traffic assignment. In Matteo Fischetti and Peter Widmayer, editors, *ATMOS 2008 - 8th Workshop on Algorithmic Approaches for Transportation Modeling, Optimization, and Systems*, Dagstuhl, Germany, 2008. http://drops.dagstuhl.de/opus/volltexte/2008/1589.

[16] A. Schöbel. Line planning in public transportation: models and methods. *OR spectrum*, 34(3):491–510, 2012.

[17] A. Schöbel and S. Scholl. Line planning with minimal travel time. In *5th workshop on algorithmic methods and models for optimization of railways*, volume 06901, 2006.

[18] L.A. Silman, Z. Barzily, and U. Passy. Planning the route system for urban buses. *Computers & Operations Research*, 1(2):201 – 211, 1974.

[19] P.J. Zwaneveld, M.T. Claessens, and N.M. van Dijk. A new method to determine the cost optimal allocation of passenger lines. In *Defence or Attack: Proceedings of 2nd TRAIL Phd Congress 1996, Part 2*, Delft/Rotterdam, 1996. TRAIL Research School.

Addendum C

J. Harbering
Delay Resistant Line Planning with a View Towards Passenger Transfers
Journal of Transportation and Logistics

Delay Resistant Line Planning with a View Towards Passenger Transfers

Jonas Harbering*[1]

[1]Institute for Numerical and Applied Mathematics,
University of Göttingen

December 11, 2015

Abstract. The line planning step, as part of the public transportation planning process, is an elementary problem. When generating public transportation systems in a conventional fashion, the line planning problem is one of the first to solve. Hence, subsequent problems rely on the solution of the line planning problem. Line planning has been studied from various perspectives and is understood very well. Still, the effect of this planning step on to the next ones has only received minor attention.

In this paper we propose a line planning model which provides a good basis for a delay resistant transportation system. To this end, the concept of preferable paths from the direct travelers line planning model is further extended. The model includes the routing of passengers in order to minimize passenger transfers. A column generation approach is shown to properly solve the proposed model. Finally, it is shown that by minimizing the passenger transfers at the line planning stage a public transportation system with high delay resistance results.

Keywords. Public Transportation Planning, Line Planning, Passenger Routing, Delay Management

1 Motivation

Public Transportation - e.g., a train system - is one of the main focuses in all societies since the basic issue of transporting people is addressed. In particular, an efficient public transportation

*The author is partially supported by the European Union Seventh Framework Programme (FP7-PEOPLE- 2009-IRSES) under grant number 246647 with the New Zealand Government (project OptALI). Also, I thank the Simulationswissenschaftliches Zentrum Clausthal-Göttingen (SWZ) for financial support.

system has to be the main goal. Efficiency is a many-faceted term in this topic (see [MBP15] for a more extensive discussion). The basic requirement of allowing people to travel between their origins and destinations has to be ensured. But efficiency may mean low traveling times of the passengers, low noise and air pollution by the system itself, an economic system which allows low fare costs, etc. Even more, the system shall not be fragile in case of disturbances.

In this work we mainly focus on the passengers' perspective. From passengers' perspective the delay of a transportation system is probably the most critical determining value. Trains being on time allow the passengers to plan their schedules detailedly around the transportation. If, additionally, the system provides low traveling times, the system will receive broader acceptance. It brings passengers from using their own vehicle to the more environmental-friendly public transportation system.

Hence, the main issue when designing a public transportation system is the requirement of low traveling times in the planned timetable *and* in the timetable under disturbances. This problem of designing public transportation systems with low planned and actual traveling times is often considered to be a problem at the stage of timetabling. In this work, we are trying to analyse the effects of the line planning, which is usually computed even before the timetabling. To this end, we prose a line planning model, which has been studied similarly in a different context and provide a solution method.

The paper is structured as follows. In the following section, literature related to different aspects of relevant planning phases of public transportation, which motivate this work, is presented. In Section 4 the model is introduced and modeling aspects are discussed. Additionally, several theoretical aspects of the presented model and an IP formulation are stated. Those results are a basis for the proposed solution approach, discussed in Section 5. Experiments showing the applicability of the method and experiments supporting the motivation idea of the model are presented in Section 6. In Section 7 the results are discussed and further lines of research are highlighted.

2 Related Literature

The planning of public transportation systems is usually split into a sequence of problems (see [CW86, BWZ97, DH03, LLER11, Sch12]). For this work the important planning phases are line planning, timetabling, and delay management which are located at levels in the planning process. They are usually solved consecutively to obtain a full-functioning public transportation system. In order to give a reasoning for the proposed approach, the problems are presented in reverse order stating relevant literature within the areas.

Delay Management The problem considered to be the last of the sequence and which is located at the operational level is the delay management problem. Given an operational timetable,

delays occur unavoidably. Those primary delays are called source delays. Once introduced into the timetable delays spread within the system due to various mechanisms. The most basic propagation mechanism is the train itself, i.e., its drivings and haltings. The drivings on the tracks and haltings in a station have certain minimal durations for all trains. The timetable specifies the actual durations which might include a time margin with respect to the minimal duration. Given a delay, it is transported from one station to another by a train if the time margin of the driving between the stations is less then the magnitude of the delay. Apart from the spreading of delays by one train also dependencies between trains exist. Passengers plan to transfer in stations from one train to another. If the first train (feeder train) is delayed, it has to be decided whether the second train (connecting train) waits or departs on time. If the train waits, the passengers can transfer and the delay propagates to the connecting train. Otherwise if the train departs on time passengers have to reschedule their plan and possibly endure a long waiting time. The decision whether to wait or to depart on time is called *wait-depart* decision. See [Sch10, SS10a] for a more detailed explanation of the propagation dependencies. Other dependencies between trains are headway or sequencing dependencies which promote the spreading even more. Additionally, vehicle or crew dependencies could be considered. For this study only passenger transfer dependencies are considered between trains and others are neglected. For a more theoretical analysis of the delay spreading see [Gov98, KK15].

The delay management problem can be divided into two different variants. In the first variant, the authors assume the delay to be known only shortly before or at the time the source delay happens. This variant is called online or real-time. Examples for such variants are given in [CHK$^+$14]. In the other variant all source delays of the planning horizon are assumed to be known in advance. This allows to state a program which decides how to deal with the delays for the entire planning horizon. Examples for this are [Sch01, Sch10, SS10a, DHSS12]. Note that for both approaches various different solution techniques exist which decide upon how to deal with the delays. In this work we are not concerned with developing a particular delay management strategy and hence focus on a simple rule widely used by practitioners [KS11]. This rule is called the fixed waiting time rule. It solves the wait-depart decisions by letting a connecting train wait for, at most, a given waiting time and otherwise lets the train depart on time.

From the stated problem and the discussed work we recognize that delays mainly spread along passenger transfers. Within the delay management problem it is dealt with by proposing different delay management strategies, such as the waiting time rule. Other such strategies can be found in [Sch10, KS11]. Additionally, many authors propose to include the consideration of potential delays in the previous planning phase, the timetabling.

Timetabling In the timetabling problem a schedule for a given a set of routes of trains and dependencies between these trains is sought. The routes of the trains consist of drivings and haltings and the dependencies are formed by passenger transfers. The drivings, haltings and

transfers are called activities. For each of the activities a lower and upper bound on its duration is given. A feasible schedule is a function assigning times to all trains at all stations such that the durations of the activities are within the given range of minimal and maximal duration. The resulting problem is a feasibility problem. Additionally, in most models the traveling times or durations are minimized.

Usually, two different models for the timetabling problem are considered. In the first model, trains are not enforced to reoccur in a periodic pattern. This model is called aperiodic. See [CCT10, LLER11] for examples. In the other model, the trains have to have a periodicity, meaning that a train reappears on the same route after a given period T. Examples for works studying this problem are [CFT02, Lie07, LM07, CT12a, SG13]. When considering the timetabling problem in this work it is referred to the latter periodic model with a period of one hour. In this work we use the method presented in [GS13b] to solve the periodic timetabling problem.

As stated before, also at this planning stage measures can be taken to combat the spreading of delays. In the timetabling problem the durations of the activities have to be at least as long as the minimal duration. At this stage one can also require additional margins or so-called buffer times which are computed as the difference between the actual and the minimal duration. Additional buffer ensure that the duration of an activity is scheduled longer than the minimal duration. Then, once a delay occurs, the buffer absorbs the delay. Many authors propose different ways of improving the robustness of timetables by including buffer times (see [FSZ09b, CT12b, GS14, APTK15] Other robustness aspects can be found in [DSCV13].

Note that including buffer in the timetable leads to the following trade off, e.g., explained in [CT12b]. If buffer times are added, the resulting timetable has to be feasible, i.e., respect these new minimal durations and hence the traveling time for all passengers increases. The more buffer is added, the more increases the planned traveling time of the passengers. On the other hand, the fewer buffer is added the smaller the delays which can be buffered. Hence, for small buffer the system is more fragile and hence the actual traveling time of the passengers increases. This trade off represents a version of the price of robustness ([BS04]).

To avoid this trade off, the idea in this work is to account for delay resistance even before the timetabling phase, i.e., in the line planning phase.

Line Planning In the line planning phase an infrastructure network consisting of nodes, e.g., stations, and edges, e.g., tracks, exists. This network is called public transportation network (PTN). For each edge of the PTN lower and upper bounds on the duration of traveling on an edge are known. Additionally, for each edge a lower and upper frequency bound is given which specify the minimal and maximal number of trains per period T. The lower frequency bounds resemble traffic count data and thereby ensure that all passengers can travel. The upper bounds reflect security or noise measures. Note that the planning period T is the same in the line planning and timetabling phase. Also, a line pool, i.e., a preselected set of possible lines, is given with

costs associated to the lines. The lines are paths in the public transportation network. The problem is to determine frequencies for all lines in the line pool such that the frequency bounds are respected. The result is a set of lines with associated frequencies, called line concept. For a general overview over different models see [BGP08, Sch12].

In the simplest model, the costs of the resulting line concept are minimized (see [CvDZ98]). A more passenger oriented model was introduced by [BKZ97] in which the number of direct passengers, i.e., passengers that do not have to transfer, is maximized. Other approaches consider the minimization of transfers (see [BGP07]) and the minimization of passengers' traveling time (see [SS06a]). An aspect of the model presented in [SS06a] is that it can be rewritten to minimize the transfers. A different approach applies a game theoretic model to the line planning in order to distribute frequencies equally among the edges (see [SS06b]).

Many different works have considered variants of the line planning problem. The effect of increasing the delay resistance at the stage on line planning has only rarely been studied. The approach in [SS06b] aims at equally distributed frequencies among the edge of the public transportation network (PTN). This maximizes the distance between either two trains and hence reduces the propagation of delays due to infrastructure capacity problems. An analysis of the effect of different line planning models on the delay management has been conducted by [GSS13].

The approach chosen in this work has the following aim. Since delays in the delay management phase spread within the system due to transferring passengers, the aim is to combat this spreading. At the stage of line planning, a realistic estimation of the paths of the passengers and hence their transfers can be developed. Based on this, the resulting line planning model minimizes the estimated number of transferring passengers. With a line concept of few transferring passengers the aim is to show that the resulting public transportation system benefits in terms of delay resistance. Note that the proposed approach can be seen similar to [BGP07]. Modeling aspects which are different in the proposed approach are highlighted in the following. Finally, the analysis of delay resistance extends the study conducted in [GSS13].

Routing of Passengers On all planning stages models which include and models which exclude the routing of passengers exist. The routing problem usually adds significant complexity to the usually already NP-hard problems [Sch14]. Hence, many authors consider models in which the routing is dealt with separately. Additionally, it should be noted, that routing at the stage of line planning – and usually also at other planning stages – are only estimations. At the stage of line planning no timetable is given and hence traveling times have to be estimated which means that also the routing is an estimation.

In the following, the important modeling parts, other existing line planning models which are important for this work and differences to the existing models are discussed.

3 Line Planning Models from Literature

Let $G = (V, E)$ be a Public Transportation Network (PTN) with nodes V and undirected edges E. For every edge the physical length and the least amount of time to pass the edge are given. Furthermore, let C be an Origin-Destination (OD) matrix with $C_{i,j}$ being the number of passengers that want to travel from $i \in V$ to $j \in V$. In order to accommodate all passengers a vehicle capacity A is enforced. Note that we assume the OD-matrix to specify the amount of passengers over the entire planning horizon of twelve hours. Similar to the passenger data the capacity is also stated for twelve hours, i.e., twelve times the actual vehicle capacity. This ensures that passenger data and vehicle capacity are fitting to each other in terms of the scale. Additionally, lower and upper frequency bounds, f^e_{min} and f^e_{max}, respectively, are given for all edges $e \in E$. The frequency bounds specify bounds on the number of trains that pass an edge in one period T. The lower frequency bound reflects traffic count data and therewith ensures that all passengers can travel in the network. The upper frequency bound represents noise and security issues. Finally, let a set of lines, the line pool $\mathscr{L}^0$, with associated costs $cost_l$ for all $l \in \mathscr{L}^0$ be given. The basic task of the line planning problem is to find frequencies $f_l \in \mathbb{N}_0$ for all lines $l \in \mathscr{L}^0$, such that

$$f^e_{min} \leq \sum_{\substack{l \in \mathscr{L}^0 \\ e \in l}} f_l \leq f^e_{max} \qquad \text{for all } e \in E \tag{1}$$

$$\tag{2}$$

The cost oriented model, introduced by [CvDZ98], minimizes the costs of the resulting line concept.

$$\left\{ \min \sum_{l \in \mathscr{L}^0} cost_l f_l \text{ — (1) is satisfied and } f_l \in \mathbb{N} \text{ for all } l \in \mathscr{L}^0 \right\} \qquad (\textbf{Cost})$$

Another model is developed implementing an operators perspective. [SS06b] propose a line planning model which aims at equally distributing the frequencies among the edges. The result is a line concept which uses the capacitated infrastructure on all the edges comparably. This problem is solved by a game theoretic approach which is shown to result in an equilibrium by solving the following model.

$$\left\{ \min \sum_{e \in E} \left(\sum_{\substack{l \in \mathscr{L}^0 \\ e \in l}} f_l \right)^2 \text{ — (1) is satisfied and } f_l \in \mathbb{N} \text{ for all } l \in \mathscr{L}^0 \right\} \qquad (\textbf{Game})$$

The next approach is passenger oriented and some of the developed ideas of that model also apply

to the model proposed in this work. Hence, this work is introduced more extensively. The idea in [BKZ97] is to maximize the number of direct travelers. In principle a very long line passing all nodes would be favorable for this objective. Since such a line is not intended due to high traveling times, the concept of preferable paths on which travelers are traveling is introduced. Let $P_G^{i,j}$ be given as a path from $i \in V$ to $j \in V$ in G. The path $P_G^{i,j}$ is assumed to be a preferable path (e.g., with a small deviation from the shortest path) or even the shortest path, with respect to the estimated traveling time or the length. If path $P_G^{i,j} \subseteq l$ it means that the entire path lies within the line $l \in \mathscr{L}^0$. Then the direct traveler model is stated by the following integer program.

$$\max \sum_{l \in \mathscr{L}^0} \sum_{\substack{i,j \in V: \\ P_G^{i,j} \subseteq l}} d_{ijl} \qquad \textbf{(Direct)}$$

$$s.t. \sum_{\substack{l \in \mathscr{L}^0: \\ P_G^{i,j} \subseteq l}} d_{ijl} \leq C_{ij} \quad \forall i,j \in V \qquad (3)$$

$$\sum_{\substack{i,j \in V: \\ e \in P_G^{i,j} \subseteq l}} d_{ijl} \leq A f_l \quad \forall e \in E, l \in \mathscr{L}^0 \qquad (4)$$

$$f_e^{min} \leq \sum_{\substack{l \in \mathscr{L}^0: \\ e \in l}} f_l \leq f_e^{max} \quad \forall e \in E \qquad (5)$$

$$d_{ijl}, f_l \in \mathbb{N} \quad \forall i,j \in V, l \in \mathscr{L}^0$$

The variables d_{ijl} for all $i,j \in V$ and all $l \in \mathscr{L}^0$ give the number of direct travelers between nodes i and j on line l. Constraint (3) ensures that at most C_{ij} passengers are traveling between i and j. In constraint (4) it is ensured that the vehicle capacity A is never exceeded for the chosen passenger paths. The upper and lower frequency bound requirements are stated in constraint (5). Finally, the objective function maximizes the number of direct travelers.

Two more relevant works further emphasize on a passenger related perspective. [BGP07] and [SS06a] present models which minimize the number of transfers and the traveling time, respectively. The proposed solution methods are column generation in [BGP07] and Dantzig Wolfe decomposition in [SS06a].

The difference of these two models to the proposed one are highlighted after stating the model of this work in the following section.

4 Line Planning with Minimal Transfers

In this section, the proposed model is introduced and the difference to existing models are highlighted. As an extension of (**Direct**) the IP-formulation is stated and further discussed.

4.1 Model

In this work the objective is to minimize the passenger transfers. The following argument from the direct travelers model (**Direct**) also holds in this case: minimizing the number of passenger transfers would result in travel paths which have only few transfers but which are very long and have high traveling times. Hence, we extend the idea of preferable paths. In (**Direct**), for every passenger from $i \in V$ to $j \in V$ one preferable path $P_G^{i,j}$ was assumed to be given. Here, we require each passenger from $i \in V$ to $j \in V$ to choose from a set of preferable paths $\mathscr{P}_G^{i,j}$ in the PTN. Let $\mathscr{P}_G$ be the union of all those sets, i.e., $\bigcup_{i,j \in V} \mathscr{P}_G^{ij} = \mathscr{P}_G$. With this notation we can now state the intended model.

(**LPT**) Given a *PTN*, $G = (V, E)$, a line pool $\mathscr{L}^0$, a set of preferable paths $\mathscr{P}_G^{ij}\ \forall i, j \in V$ on G, an $OD-$matrix C and the vehicle capacity A. Find a feasible line concept $(\mathscr{L}, f)$ such that (1) is satisfied and the number of passenger transfers is minimized.

Since the actual passenger transfers can only be determined once the timetable is known the transfers at the line planning stage can only be an estimation. For this estimation we use the concept of the Change&Go-Graph (CG-Graph) [SS06a]. Note that the term *change* corresponds to the term *transfer* in this work.

Definition 4.1. Let a PTN $G = (V, E)$ and a set of lines $\mathscr{L}$ be given. The *Change&Go*-Graph $CG = (\mathscr{V}, \mathscr{D})$ is given as

$$\begin{aligned}
\mathscr{V} &= \{(v, l) : v \in V, l \in \mathscr{L} \text{ and } v \in l\} \cup \{(v, 0) : v \in V\} \\
\mathscr{D} &= \mathscr{D}_{line} \cup \mathscr{D}_{transfer} \cup \mathscr{D}_{OD}, \\
\mathscr{D}_{line} &= \{((v_1, l), (v_2, l)) : (v_1, v_2) \in l, v_1, v_2 \in V, l \in \mathscr{L}\} \\
\mathscr{D}_{transfer} &= \{((v, l_1), (v, l_2)) : v \in l_1 \cap l_2, v \in V, l_1, l_2 \in \mathscr{L}\} \\
\mathscr{D}_{OD} &= \{((v, 0), (v, l)) : v \in l, v \in V, l \in \mathscr{L}\}
\end{aligned}$$

Note that arcs from $\mathscr{D}$ are undirected. For usability some notation is introduced.

Notation 4.2. In some cases we may write $(e, l) \in \mathscr{D}_{line}$ which means that (e, l) represents $((v_1, l), (v_2, l))$ or $((v_2, l), (v_1, l))$ with $e = (v_1, v_2)$.

We can now depict the relation of paths in the PTN and on the Change&Go-Graph.

Notation 4.3. Given a set of preferable paths $\mathscr{P}_G^{ij}$, a passenger traveling from $i \in V$ to $j \in V$ can be routed in CG from $(i, 0) \in \mathscr{V}$ to $(j, 0) \in \mathscr{V}$. Let $p \in \mathscr{P}_G^{ij}$ be a path on G, then the path is described as

$$p = (v_1, e_1, \ldots, v_{t_p - 1}, e_{t_p - 1}, v_{t_p}),$$

where $v_1, \ldots, v_{t_p} \in V$ and $e_i = (v_i, v_{i+1}) \in E$ for all $i = 1, \ldots, t_p - 1$. t_p is the number of passed nodes in G. Even more, we have $v_1 = i$ and $v_{t_p} = j$. This path does not specify which lines are used. A path $\mathfrak{p}$ realizing p on CG can then be described as

$$\mathfrak{p} = (\mathfrak{v}_1, \mathfrak{d}_1, \mathfrak{v}_2, \mathfrak{d}_2, \ldots, \mathfrak{v}_{r_\mathfrak{p}-1}, \mathfrak{d}_{r_\mathfrak{p}-1}, \mathfrak{v}_{r_\mathfrak{p}}),$$

where $\mathfrak{v}_1, \ldots, \mathfrak{v}_{r_\mathfrak{p}} \in \mathscr{V}$, $\mathfrak{d}_i = (\mathfrak{v}_i, \mathfrak{v}_{i+1}) \in \mathscr{D}$ for all $i = 1, \ldots, r_\mathfrak{p} - 1$. $r_\mathfrak{p}$ is the number of passed nodes in CG. In order to represent p on CG we require

$$\mathfrak{v}_1 = (i, 0), \mathfrak{v}_{r_p} = (j, 0) \text{ and } \{e \in E \mid (e, l) \in \mathfrak{p}\} = \{e_1, \ldots, e_{t_p-1}\}$$

We thus define the projection of $\mathfrak{p}$ on G as $\mathfrak{p}|_G = p$. Similarly, we say that $\mathfrak{p}$ is a realization of p on CG. Let $\mathscr{P}_{CG}^{ij}$ be the set of all possible realizations of paths from $\mathscr{P}_G^{ij}$ on CG. For every possible path $\mathfrak{p} \in \mathscr{P}_{CG}^{ij}$ the number of used transfers $c_\mathfrak{p}$ can be calculated as the number of used arcs from $\mathscr{D}_{transfer}$, i.e., $c_\mathfrak{p} = |\{\mathfrak{d} \in \mathfrak{p} \ : \ \mathfrak{d} \in \mathscr{D}_{transfer}\}| \geq 0$. By this construction the number of passenger weighted transfers can be minimized.

We will discuss the number of paths in Section 4.3 in more detail.

4.2 Comparison to Existing Models

Comparing the model used in this work to [BGP07] and [SS06a], the following differences are found. In both models from [BGP07, SS06a] all possible paths are allowed for passengers. It may mean, that passengers take long detours in a solution. In [BGP07] it is noted that restricting the paths travel time or length significantly increases the complexity of the problems. In this work, as introduced in Section 4, the concept of preferable paths is further extended from [BKZ97]. This allows to specify only short or preferable paths previous to the optimization. It ensures that the proposed method can deal with the resulting complexity. Another main difference to [BGP07] is the generation of lines. In [BGP07] is modeled that lines are generated within the line planning problem. In this work the line planning problem is assumes a line pool to be given. An example for computing the line pool is given in [GHS15]. Furthermore, in this work we show the applicability of the proposed method on considerably larger instances compared to both [BGP07] and [SS06a].

In the following an IP-formulation for solving (**LPT**) is developed.

4.3 IP-Formulation

The construction of the CG-graph CG relies on the line set $\mathscr{L} = \mathscr{L}^0$. We now outline the development of our IP-formulation based on the formulation used by [BKZ97]. Due to the maximization in (**Direct**), the number of passengers traveling from $i \in V$ to $j \in V$ has to be

bounded from above in constraint (3). Since in our new model we aim to minimize the passenger weighted transfers, the direction of condition (3) changes. Also constraint (1) is present in (**Direct**) since the program only routes passengers who do not have to transfer. For passengers that need to transfer the lower frequency bound has to remain in the program. In our model all passengers are routed on preferable paths. Hence, the intention of the lower frequency bound is satisfied by supplying sufficient capacity and we can drop the lower bound of constraint (1). The IP-formulation is then given by

$$(\textbf{IP-LPT}) \qquad \min \sum_{i,j \in V} \sum_{p \in \mathscr{P}_{CG}^{ij}} d_p c'_p \tag{6}$$

$$\sum_{p \in \mathscr{P}_{CG}^{ij}} d_p \geq C_{ij} \quad \forall i,j \in V \tag{7}$$

$$\sum_{i,j \in V} \sum_{\substack{p \in \mathscr{P}_{CG}^{ij} \\ (e,l) \in p}} d_p \leq A f_l \quad \forall l \in \mathscr{L}^0, \forall e \in l \tag{8}$$

$$\sum_{\substack{l \in \mathscr{L}^0 \\ e \in l}} f_l \leq f_e^{max} \qquad \forall e \in E \tag{9}$$

$$d_p \in \mathbb{N}_0 \qquad \forall p \in \mathscr{P}_{TG} \tag{10}$$

$$f_l \in \mathbb{N}_0 \qquad \forall l \in \mathscr{L}^0 \tag{11}$$

Let $p \in \mathscr{P}_{CG}^{ij}$ be an $i-j-$path in CG then d_p specifies the number of passengers traveling on that path. The variable f_l determines the frequency of line $l \in \mathscr{L}^0$. The parameter c_p equals the number of transfers on path p. Constraints (7), (8) and (9) are similar or equal to the corresponding ones of (**Direct**). The objective function (6) then minimizes the number of passenger weighted transfers.

We direct towards constraint (7) which ensures that every passenger is indeed considered in the routing.

Lemma 4.4. *There exists an optimal solution to (**IP-LPT**) which satisfies (7) with equality.*

Proof. Let d^*, f^* be an optimal solution to (**IP-LPT**). If constraint (7) is satisfied with equality for all $i,j \in V$, there is nothing to show. Hence, let us assume there exists a combination $i,j \in V$ such that constraint (7) evaluates to a strict inequality, i.e.,

$$\sum_{p \in \mathscr{P}_{CG}^{ij}} d_p > C_{ij} \tag{12}$$

We will now construct a new optimal solution such that equality is obtained for constraint (7).

For all $p \in \mathscr{P}_{TG}^{ij}$ with $c_p = 0$ we can reduce the value of d_p until either equality is obtained in constraint (7) or $d_p = 0$. The reducing of the values d_p does not provoke any conflict with

neither constraint (8) nor (9). The resulting solution d^+, f^+ has same objective value as d^*, f^* and is still feasible, thus optimal. If d^+, f^+ satisfies constraint (7) with equality, a desired optimal solution is found.

Now we will show that if equality is still not obtained for d^+, f^+, it has not been optimal. Since constraint (7) still evaluates to a strict inequality, we can reduce d_p for any arbitrary $p \in \mathscr{P}_{TG}^{ij}$, where $d_p > 0$ and $c_p > 0$. The reducing does still not provoke any conflict with neither constraint (8) nor (9). Hence, we can reduce the objective value and still obtain a feasible solution which is a contradiction to the optimality of the solution d^*, f^*. □

The following theorem shows the complexity of (**LPT**).

Theorem 4.5. *(LPT) is NP-complete, even if we have only one OD-pair.*

Proof. We reduce the problem *set partitioning* (**SP**) to (**LPT**). An instance of (**SP**) is given by the sets $\mathscr{M} = \{m_1, \ldots, m_n\}$ and $\mathscr{N} \subseteq 2^{\mathscr{M}}$. The aim is to find a set $\bar{N} \subseteq \mathscr{N}$ such that for all $m \in \mathscr{M}$ $\exists!\ N \in \bar{N}\ :\ m \in N$.

For the reduction, define

$$
\begin{aligned}
&\text{PTN } G = (V, E)\\
&\qquad V = \{u, v\} \cup \{t_m \mid m \in \mathscr{M}\}\\
&\qquad E = \{(u, t_m), (t_m, v) \mid m \in \mathscr{M}\}\\
&\qquad \mathscr{L}^0 = \{l_N \mid N \in \mathscr{N}\}\ \text{, where } l_N = \{(u, t_m), (t_m, v) \mid m \in N\}\ \text{for all } N \in \mathscr{N}\\
&\qquad C_{i,j} = \begin{cases} |\mathscr{M}| & \text{, i=u, j=v}\\ 0 & \text{, otherwise.}\end{cases}\\
&\quad f_e^{max} = 1 \text{ for all } e \in E\\
&\qquad A = 1\\
&\quad \mathscr{P}_G^{ij} = \{(u, (u, t_m), t_m, (t_m, v), v) \mid m \in \mathscr{M}\}\ \text{for } i = u, j = v
\end{aligned}
$$

See Figure 1 for an example graph.

Then the following holds. There exists a feasible solution to (**SP**) if and only if there exists a feasible solution to (**LPT**).

"⇒" Let F be a solution to (**SP**), meaning that $F \subseteq \mathscr{N}$ and for all $m \in \mathscr{M}$ $\exists! N \in F\ :\ m \in N$. Let $f_{l_N}^* = 1$ for all $N \in F$ and 0 otherwise. Furthermore let $d_p^* = 1$ for all $p \in \mathscr{P}_{CG}^{uv}$, where $((u, t_m), l_N), ((t_m, v), l_N) \in p$ for $m \in N$ and $N \in F$. For all other $p \in \mathscr{P}_{CG}^{ij}$ let $d_p^* = 0$. This is well defined, i.e., there exists only one $p \in \mathscr{P}_{CG}^{uv}$ such that $((u, t_m), l_N), ((t_m, v), l_N) \in p$ for $m \in N$ and $N \in F$, since for all $m \in \mathscr{M}$ $\exists! N \in F\ :\ m \in N$.

We now have to show the feasibility of f^*, d^*. Due to the construction of d^* we have that

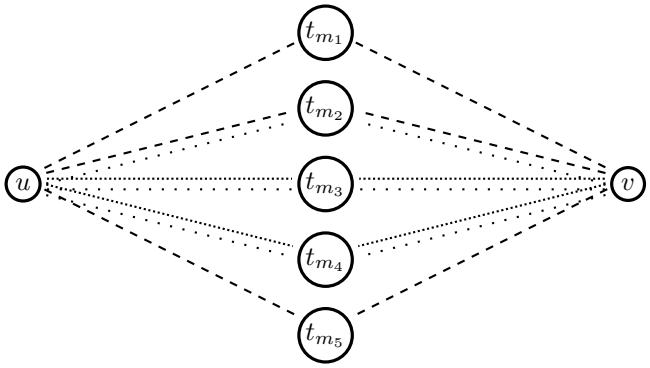


Figure 1: Sketch of graph G with $\mathcal{N} = \{\{m_1, m_2, m_5\}, \{m_2, m_3, m_4\}, \{m_3, m_4\}\}$.

$\sum_{p \in \mathcal{P}_{CG}^{uv}} d_p^* = C_{uv}$. Additionally, $C_{ij} = 0$ for all $i \neq u$ and $j \neq v$. Hence, constraint (7) is fulfilled.

Since for all $m \in \mathcal{M}$ there exists exactly one $N \in F$ such that $m \in N$ we also have that for all $e \in E$ there exists only one $l_N \in \mathcal{L}^0$ with $f_{l_N}^* > 0$ ($f_{l_N}^* = 1$) such that $e \in l_N$. This means $\sum_{\substack{l \in \mathcal{L}^0 \\ e \in l}} f_l^* = 1 \leq f_e^{max}$ and hence the constraint (9) is fulfilled for each $e \in E$. Finally, due to the feasibility of F and hence the construction of d^* we also obtain $\sum_{\substack{p \in \mathcal{P}_{CG}^{uv} \\ (e, l_N) \in p}} d_p^* = 1$ if $N \in F$. But then also $f_{l_N} = 1$. Otherwise if $N \notin F$ we have that $f_{l_N} = 0$ but then also $\sum_{\substack{p \in \mathcal{P}_{CG}^{uv} \\ (e, l_N) \in p}} d_p^* = 0$. It means that (8) is fulfilled. Hence, the constructed solution d^*, f^* is feasible.

"$\Leftarrow$" Let us now assume a solution f^*, d^* is given for (**LPT**). Define $F := \{N \in \mathcal{N} : f_{l_N} = 1\}$. We now have to show that every edge is only traversed by exactly one line. We have that

$$1 = \frac{\sum_{\substack{l \in \mathcal{L}^0 \\ e \in l}} \sum_{\substack{p \in \mathcal{P}_{CG}^{uv} \\ (e,l) \in p}} d_p^*}{A} \leq \sum_{\substack{l \in \mathcal{L}^0 \\ e \in l}} f_l \leq f_e^{max} = 1, \tag{13}$$

where the first equality of (13) holds since $A = 1$, $f_l \in \{0, 1\}$ and constraints (8) and (9) hold. Hence we have that $\sum_{\substack{l \in \mathcal{L}^0 \\ e \in l}} f_l = 1$ which means that each edge is only traversed by exactly one line. Since the edges relate to the elements of $\mathcal{M}$ and the lines relate to the set $\mathcal{N}$ we find that $\forall\, m \in \mathcal{M} : \exists! N \in F \ : \ m \in N$.

□

More complexity results on special graphs and with particular OD-information can be found in [SS10b, Sch14]. We now direct to the complexity of the entire IP-formulation expressed by the number of variables.

Lemma 4.6. *The number of variables may be exponential in the number of nodes $|V|$. This is even true if for all $i, j \in V$ the preferable path set $\mathcal{P}_G^{ij}$ only contains one element.*

Proof. It is sufficient to show that for one (i,j)-combination of the number of paths $|\mathcal{P}_{CG}^{ij}|$ is of exponential order in the number of nodes $|V|$. Suppose $|\mathcal{P}_{G}^{ij}| = 1$, i.e., there is only one preferable $i-j-$path in G. Assume that m stations are visited on the shortest path between i and j, not including i and j. Then, at each station a passenger can have a transfer. This transfer can occur between every pair of lines from $\mathscr{L}^0$. First, we can think of splitting the number of paths up by looking at all paths that have exactly $0 \leq c \leq m$ transfers. The highest number of paths is clearly obtained, when all lines from $\mathscr{L}^0$ give service from i to j. Thereby, we do not consider taking multiple transfers at the same station. First take $c = 0$. Then the number of possible paths is clearly $\left|\mathscr{L}^0\right|$. Now consider $c = 1$. Then the passenger has the choice of taking either of the lines from $\mathscr{L}^0$. Subsequently the transfer may occur at any of the next m stations. The transfer takes place by changing from the actual train to any available which are $\left|\mathscr{L}^0\right| - 1$. Thus, we have for $c = 1$ a number of

$$\left|\mathscr{L}^0\right| \binom{m}{1} \left(\left|\mathscr{L}^0\right| - 1\right)$$

Thus, when separating for the number of transfers $c = 1, \ldots, m$, we have,

$$|\mathcal{P}_{CG}^{ij}| = \sum_{c=0}^{m} \binom{m}{c} \left|\mathscr{L}^0\right|^{1+c} = \left|\mathscr{L}^0\right| \left(1 + \left|\mathscr{L}^0\right|\right)^m \in \mathcal{O}\left(\left|\mathscr{L}^0\right|^{m+1}\right)$$

Since m, the number of stops between two stations i and j, is at most $|V| - 2 = n - 2$, the complexity evaluates to $\mathcal{O}\left(|\mathscr{L}^0|^{n-1}\right)$ □

Lemma 4.7. *Suppose $|\mathcal{P}_{G}^{ij}| = 1$ for all $i \neq j \in V$ and a maximal number of transfers on each path $p \in \mathcal{P}_{TG}$ is fixed to s then the number of variables is of order $\mathcal{O}\left(n^{s+2}\right)$.*

Proof. This follows directly from the proof of 4.6 by summing up c until s only.

$$|\mathcal{P}_{CG}^{ij}| = \sum_{c=0}^{s} \binom{m}{c} \left|\mathscr{L}^0\right|^{1+c} \leq \left|\mathscr{L}^0\right|^{1+s} \sum_{c=0}^{s} \binom{m}{c} \leq \underbrace{\left|\mathscr{L}^0\right|^{1+s}}_{\text{const.}} (m+1)^s \in \mathcal{O}\left(m^s\right)$$

Now, m can at most be $n-2$, meaning that between i and j exactly all other $|V|-2 = n-2$ nodes have to be passed on the shortest path in G. Even more there exist $n(n-1)$ different combinations of $i, j \in \{1, \ldots, n\}, i \neq j$. Thus, we obtain a number of $n(n-1)(n-2)^s \left|\mathscr{L}^0\right|^{s+1} \in \mathcal{O}\left(n^{s+2}\right)$. □

Even though the number of paths and thus variables is of polynomial size if the number of transfers are bounded, this might still be a very large number. In the next section we will discuss a method to tackle problems with a high number of variables, namely column generation.

5 Solution Method

Based on the IP formulation (**IP-LPT**) we present a column generation approach. The column generation procedure decomposes the IP into a master program and a subproblem. The master program is the LP relaxation of (**IP-LPT**) with only a narrowed number variables. The subproblem determines attractive variables to add to the master program. We here omit a detailed discussion on the column generation procedure and refer to a general description on the functioning of column generation in [BJN$^+$98].

The column generation procedure needs a feasible starting solution, discussed in the following section. From a feasible solution attractive additional variables can be generated in a subproblem which is described in Section 5.2. Once the column generation stops the final solution only solves the LP relaxation of (**IP-LPT**). In Section 5.3 it is discussed how integer solutions are obtained from that.

5.1 Starting Tableau

Since $\mathscr{L}^0$ is assumed to be given in the input, the lines do not have to be generated. Hence, we consider all lines from $\mathscr{L}^0$ in the master program. In contrast, the number of passenger paths is too large to explicitly state all of them in the master program. It means that in a first step a set of passenger paths has to be computed which admits a feasible starting solution. It holds that the better the starting solution the fewer pricing steps have to be undergone and thus fewer time is needed to solve the model.

In this approach we use a heuristic which, for each path $p \in \mathscr{P}_G^{ij}$, computes k paths on a graph similar to CG, called *Transfer-Graph*. These k paths are computed as the k shortest in terms of the number of transfers which represent p on CG. Hence, the problem solved is a k-shortest path problem on the Transfer-Graph. Previous to the formal definition of the Transfer-Graph we introduce a notation.

Notation 5.1. Let $p = (v_1, e_1, \ldots, e_{t_p-1}, v_{t_p}) \in \mathscr{P}_G^{ij}$ be a path and $l \in \mathscr{L}^0$ a line. By adding all edges from l to the vertex set $\{v_1, \ldots, v_{t_p}\}$ and removing those vertices which have degree 0, we obtain connected components $l^1 \ldots, l^q$ for some number $q \in \mathbb{N}$.

The connected components are itself sublines, i.e., parts of lines. With this understanding we can define the deviated Transfer-Graph.

Definition 5.2. Let $p = (v_1, e_1, \ldots, e_{t_p-1}, v_{t_p}) \in \mathscr{P}_G^{ij}$ be the path on which the Transfer-Graph T_p

is based. Then the Transfer-Graph T_p is constructed as

$$
\begin{aligned}
T_p &= (\mathscr{V}', \mathscr{D}') \\
\mathscr{V}' &= \mathscr{V}^1 \cup \mathscr{V}^2 \\
\mathscr{V}^1 &= \{v \mid v \in p\} \\
\mathscr{V}^2 &= \left\{ l_j^i \mid l_j \in \mathscr{L}^0, 1 \leq i \leq q \right\} \\
\mathscr{D}' &= \left\{ (v, l_j^i) \mid v \in l_j^i, v \in \mathscr{V}^1, l_j^i \in \mathscr{V}^2 \right\}
\end{aligned}
$$

Example 5.3. Figure 2 depicts an example for the construction of a Transfer-Graph. Note that the graph in the left of Figure 2 is only a part of the infrastructure network. The graph is a subgraph induced by the edges of path $p = (v_1, (v_1, v_2), v_2, \ldots, v_4, (v_4, v_5), v_5)$ with three possible lines l_1, l_2, l_3. The line l_1 passes the vertices v_1, v_2, v_3 and v_4. Line l_2 passes the vertices v_2, v_3 and v_4. Line l_3 passes the vertices v_1 and v_2, followed by other vertices of the infrastructure network, and after that it passes the vertices v_3, v_4 and v_5.

As an intermediate step in order to construct the Transfer-Graph, the edges of, e.g., line l_3 are added to the vertex set $\mathscr{V}^1 = \{v_1, \ldots, v_5\}$. This results in two connected components l_3^1 and l_3^2. In a similar way the connected components l_1^1 and l_2^1 are obtained, thus $\mathscr{V}^2 = \{l_1^1, l_2^1, l_3^1, l_3^2\}$.

Then the Transfer-Graph is constructed. For each connected component a vertex is added to the vertex set $\mathscr{V}^1 = v_1, \ldots, v_5$. If a vertex from $\mathscr{V}^1 = \{v_1, \ldots, v_5\}$ belongs to a connected component an edge is drawn between the corresponding vertices. Thus the set $\mathscr{D}'$ is obtained. Thus, the right of Figure 2 is obtained.

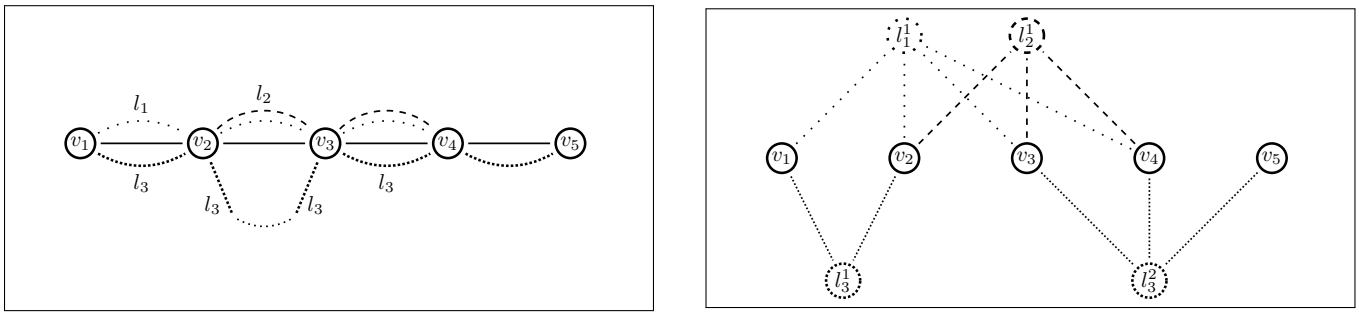


Figure 2: Construction of a Transfer-Graph

On this graph the number of transfers of an $i-j-$path $p \in \mathscr{P}_{TG}^{ij}$ is calculated via the number of passed vertices of set $\mathscr{V}^2$ The number of transfers is one less than the number of passed vertices from set $\mathscr{V}^2$. The reason is that a path contains one transfer less than it uses lines.

The Transfer-Graph omits a special structure.

Lemma 5.4. *Transfer-Graphs are convex bipartite.*

Proof. First we find that $\mathscr{V}' = \mathscr{V}^1 \cup \mathscr{V}^2$ and $\mathscr{V}^1 \cap \mathscr{V}^2 = \emptyset$. From the definition of $\mathscr{D}'$ we also obtain that Transfer-Graphs are bipartite, i.e., for all $a = (u, v) \in \mathscr{D}'$ we have that either $u \in \mathscr{V}^1$ and $v \in \mathscr{V}^2$ or vice versa. Let $N_T(v)$ be the neighborhood of a vertex $v \in \mathscr{V}^2$, which is the set of all vertices adjacent to v. We only have to show that there exists a bijective mapping $f : \mathscr{V}^1 \to \{1, \ldots, |\mathscr{V}^1|\}$, such that for all $v \in \mathscr{V}^2$ and for any two vertices $u, w \in N_T(v)$ there exists no $z \notin N_T(v)$ such that $f(u) < f(z) < f(w)$.

We define the function f by

$$v \mapsto \text{ position of } v \text{ in path } p$$

Since the vertices $l_j^i \in \mathscr{V}^2$ are defined via the connected components when adding line $l_j \in \mathscr{L}^0$ to $\{v_1, \ldots, v_{t_p}\}$, we obtain that f defines the required bijective mapping. □

The k-shortest-path problem on the Transfer-Graph results in a complexity of $\mathcal{O}\left(k|\mathscr{V}'|\left(|\mathscr{D}'|\right.\right.$ $\left.\left.+|\mathscr{V}'|log|\mathscr{V}'|\right)\right)$ ([Yen71]).

Once a feasible starting solution is found, the column generation procedure of repeatedly solving the master and the subproblem can be commenced. The following section describes the subproblem which results.

5.2 Pricing of Path Variables

Let π and γ be the dual variables corresponding to constraint (7) and (8), respectively. Furthermore, let

$$D_{((i,j),p)} = \begin{cases} 1 & \text{, if path } p \text{ from } i \text{ to } j. \\ 0 & \text{, otherwise.} \end{cases} \qquad D \in \{0,1\}^{|V|^2 \times |\mathscr{P}_{CG}|}$$

$$L^{CG}_{((e,l),p)} = \begin{cases} 1 & \text{, if } (e,l) \in p. \\ 0 & \text{, otherwise.} \end{cases} \qquad L^{CG} \in \{0,1\}^{|\mathscr{D}_{line}| \times |\mathscr{P}_{CG}|}$$

With this notation the pricing of the path-related variables can be stated as

$$\begin{aligned} c_p - (D_p)^t\pi + (L_p^{CG})^t\gamma &= -\sum_{k,l \in V} D_{((k,l),p)}\pi_{(k,l)} + c_p + \sum_{(e,l)\in\mathscr{D}_{line}} L^{CG}_{((e,l),p)}\gamma_{(e,l)} \quad \text{for some } p \in \mathscr{P}_{ij}^{CG} \\ &= -\pi_{(i,j)} + c_p + \sum_{(e,l)\in p} \gamma_{(e,l)} \end{aligned} \tag{14}$$

Here, the parameters π and γ are known from the dual of (**LPT**). What we are looking for is the path p and we want to minimize (14). This is a shortest path problem in CG with edge weights $\gamma_{(e,l)} \geq 0$ on all edges $(e, l) \in \mathscr{D}_{line}$, described in the following. More explicitly, weights

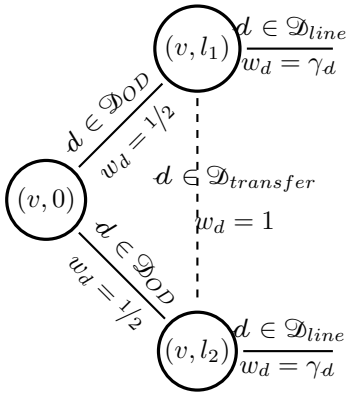


Figure 3: Example for transfers on a Transfer&-Go Graph

are assigned to arcs in order to obtain the required assessment, i.e.,

$$w_d = \begin{cases} \gamma_{(e,l)} & , (e,l) = d \in \mathcal{D}_{line} \\ \frac{1}{2} & , d \in \mathcal{D}_{OD} \\ 1 & , d \in \mathcal{D}_{transfer} \end{cases}$$

Hence a path p in CG has the weight

$$w_p = \sum_{\substack{d \in \mathcal{D}_{OD} \\ d \in p}} \frac{1}{2} + \sum_{\substack{d \in \mathcal{D}_{transfer} \\ d \in p}} 1 + \sum_{\substack{(e,l) \in \mathcal{D}_{line} \\ (e,l) \in p}} \gamma_{(e,l)} = 1 + c_p + \sum_{(e,l) \in p} \gamma_{(e,l)}$$

Thus, for all $i, j \in V$ we seek for the path p where $w_p - \pi_{(i,j)}$ is minimized, i.e.,

$$p^* = \underset{\substack{i,j \in V \\ p \in \mathcal{P}_{TG}^{ij}}}{\operatorname{argmin}} w_p - \pi_{(i,j)} - 1$$

Note that transfers may actually be taken either along a transfer arc or along two consecutive arcs from $\mathcal{D}_{OD}$. Indeed, we can speed up the shortest path search by neglecting the transfer arc set $\mathcal{D}_{transfer}$. See Figure 3 as an illustration of this property.

If the minimum value $w_{p^*} - \pi_{(i,j)} - 1 < 0$, the path p^* is added to the primal program. Otherwise, all reduced costs are proven to be non-negative and hence the primal solution is optimal, i.e., the algorithm stops. As the solution is only a solution to the LP relaxation of (**IP-LPT**), the following section discusses how to obtain integer values.

5.3 IP-Solution

In the above discussed column generation only the LP-relaxation of (**LPT**) is solved. This leaves us with the final problem of determining an integer solution for (**LPT**) from a solution to the relaxed problem where all variables are real numbers.

It is shown that if the matrix of the LP relaxation is totally unimodular (TU) or has the total dual integrality (TDI) property, the solution turns out to be integer. See [BW05] for further explanation on these properties. Neither of these properties is fulfilled by the matrix of the LP relaxation of (**IP-LPT**). Hence, another approach is employed. In the literature different exact and heuristic approaches are known. Examples for exact methods are, e.g., branch and bound ([DL05]) and branch and price (see [BJN$^+$98]) procedures. To this end, also commercial solvers as Xpress, Cplex or Gurobi which implement general branch and bound frameworks, can be used. For heuristics a wide range of methods exist, some of which are greedy, randomized rounding based or local search (see [JMS$^+$10]). Mostly, approaches are chosen which are specifically designed for the considered problem.

In this work we set up a mixed integer program (MIP) which corresponds to the final primal program of the column generation. From the column generation we obtain an optimal solution to the relaxed problem. Now those path variables are added to the MIP-formulation which are attractive. The assessment of the attractive variables comes the optimal solution to the relaxed problem which specifies the reduced costs for every variable. Only those variables are added to the MIP which have reduced costs no bigger than a certain threshold value w, usually one or two. Since the line planning step is in practice taken long time before its solution is realized it is sufficient to know the number of passengers as real values. In the subsequent planning stages the paths of the passengers are computed more precisely. Hence, in the MIP we leave the passenger path variables relaxed and only constrain the frequency variables to be integer.

In the experiment, the MIP formulation is solved by Xpress. Note that this approach neither guarantees to find a feasible nor an optimal solution. Still, we use this approach as the experiments show that good and even optimal solutions are obtained reliably.

6 Experiments

In this section the experiments showing the applicability of the method and the delay resistance of the line concepts resulting from (**LPT**) are given. Note that all computations are executed with the LinTim software package (see [GHS, GSS13]). The process chart depicting the functionalities of LinTim is given in Figure 4.

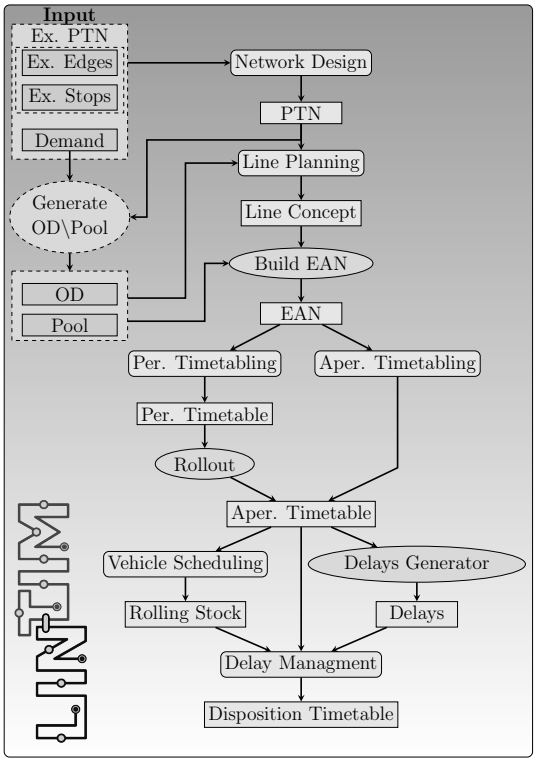


Figure 4: LinTim computation scheme for public transport planning

Name	$\|V\|$	$\|E\|$	$\|\mathscr{L}^0\|$	$\|C\|$
toy	8	8	8	44
lowersaxony	34	35	30	306
bahn	250	326	132	6106

Table 1: Table of test data sets

6.1 Results on Applicability of Method

The applicability of the proposed method is shown on small network instances. For (**LPT**) we compare the objective value of the solution and the computation time of the integer program to the objective value of the solution and the computation time of the column generation approach.

Data Set The data set instances for this experiment are toy and lowersaxony. The sizes of these instances can be gathered from Table 1. The toy data set is a small invented data set and the lowersaxony data set is derived from the small real world network of Lower Saxony, Germany. Note that in the lowersaxony data set only OD-pairs are considered for which the shortest path in the PTN is at most 5 edges long. This decreases the computation time such that instances based on lowersaxony can be solved optimally by the integer program.

Experiments The column generation approach is compared with the IP formulation. We use the data sets toy and lowersaxony, each with a certain amount of OD-pairs, increasing in 10-percent steps, and solved the problem with both approaches. The results are depicted in Figure 5. The instance "toy;10" hereby specifies that the data set toy and 10% of the OD-pairs are considered. All other instances are declared likewise.

Evaluation Parameters The solutions quality of (**IP-LPT**) and the column generation approach are compared in terms of the objective value, i.e., the number passenger transfers, and the computation time. As the solution to (**IP-LPT**) is optimal, the quality of solutions from the column generation approach can be judged with respect to the optimal objective value.

Results The results of this experiment are depicted in Figure 5. For all instances generated from the toy data set both approaches lead to the same results, both in terms of the objective value and in terms of the computation time. The results on instances generated from lowersaxony differ recognizably. The objective values of solutions obtained from the column generation approach have the same values as from the IP formulation. Hence, we can conclude that the column generation approach finds the optimal solution in all tests on the data sets toy and lowersaxony. Comparing the computation times of the IP formulation and the column generation we recognize large differences. The computation times of the column generation do not show any effect with increasing instance sizes. In contrast, the computation times of the IP formulation are increasing

drastically with increasing instance sizes. This is also the reason why larger data sets can eventually not be solved by the IP formulation and only the column generation approach provides solutions. Hence, from these experiments, we can conclude that the column generation provides good solutions in a short amount of time.

Note that this experiment does not mean that the column generation approach always leads to optimal solutions. It is only meant to be an indication that the solutions provided are reasonably good with respect to the optimal solution.

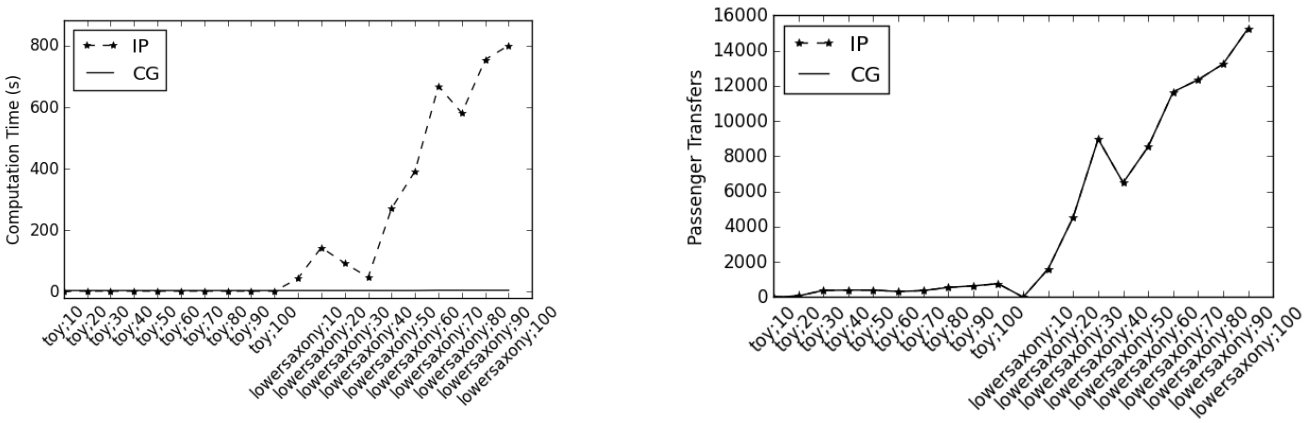


Figure 5: Comparing solutions of IP and CG for computation time and solution quality

6.2 Results on Delay Resistance

Directing to the delay resistance claim, we computed our results on the more realistic bahn data set.

Data Set For the data set in this experiment a realistic instance derived from the German high speed railway network is derived. Hence, it reflects a realistically shaped and sized instance. See Table 1 for details on the instance size.

Experiments From the previous experiments we recognize that the IP formulation (**IP-LPT**) can only be solved for small instances. The bahn instance is far too big to be able to compute the IP formulation, hence it can only be solved by the column generation approach. In order to test the delay resistance of the presented model we follow the scheme depicted in Figure 4. The following stepwise computation, similar to [GSS13], is executed.

Line Planning. We will compare the results of the model (**LPT**) to the results of the models (**Cost**) from [CvDZ98], (**Direct**) from [Die78, BKZ97] and (**Game**) from [SS06b, SS13].

The model (**LPT**) is based on a set of preferable paths $\mathscr{P}_G$. For each set of preferable paths $\mathscr{P}_G^{ij}$ for $i, j \in V$ the ten shortest $i-j$-paths in terms of the number of edges are computed. Since

they are usually not unique, the model (**LPT**) is computed four times, each time with a different set $\mathscr{P}_G$. In order to compute a feasible starting solution, ten realizing paths $p \in \mathscr{P}_{TG}^{ij}$ on CG are computed for every path $p \in \mathscr{P}_G^{ij}$ on the PTN (see Section 5.1). After some pricing steps all reduced costs are non-negative (see Section 5.2). Then, all variables that have reduced costs less or equal to two ($w = 2$) are added to (**IP-LPT**) and the problem is solved as a MIP (see Section 5.3). The Xpress-internal branch and bound procedure solving (**IP-LPT**) stops if the gap between the objective value and the best known lower bound is at most 1% (see [Opt07]).

Routing of Passengers at the Stage of a Line Concept (RLC). After the line planning, all passengers given in the OD-matrix are routed along their shortest paths in the PTN. Only edges can be used which are covered by the line concept. The edge weights are given as the physical length of the edges. This step is only needed for evaluation purposes.

Converting Line Planning Output to Timetabling Input. Once a line concept is obtained, an event-activity network is computed (see [Sch10]). The event-activity network $\mathscr{N} = (\mathscr{E}, \mathscr{A})$ consists of events $\mathscr{E}$ and activities $\mathscr{A}$. The events reflect all arrivals and departures of all trains at all stations. The activities are the train driving and halting activities and the passenger transfers. For these activities the lower and upper bounds on the durations are previously given. See [Nac98] for a more explicit construction.

Routing of Passengers on the Event-Activity Network (REAN). First, the passengers of the OD-matrix are distributed among all possible departure times. The activity weights are the mean of the lower and upper duration for driving and halting activities, and $\frac{T}{f_{l_i} f_{l_j}}$ for a transfer between line l_i of frequency f_{l_i} and line l_j of frequency f_{l_j}, $l_i, l_j \in \mathscr{L}$. Then passengers are routed along their shortest paths in the event-activity network.

Timetabling. The event-activity network is an input for the periodic timetabling problem ([GS13a]), which is solved next. As outlined in Section 2, the periodic timetable assigns times to all events from $\mathscr{E}$. In this work we use the algorithm proposed in [GS13b] which minimizes the estimated traveling time of the passengers. It reformulates the periodic timetabling problem as a constraint satisfaction problem, finds a feasible solution and improves on that with a heuristic algorithm.

Routing of Passengers based on a Timetable (RT). Again, passengers are distributed among all possible departure times. Their shortest path in the event-activity network is calculated based on the traveling times specified in the timetable.

Delay Management. For the delay management, the timetable is rolled out over a planning horizon of twelve hours (twelve periods). It means that the periodic event-activity network together with the timetable are converted into a time-expanded network covering twelve periods. Into this time-expanded timetable a set of 100 delays (on any activity) of a random magnitude between 60 and 900 seconds are introduced. The set of delays introduced into the timetable is

called a *delay scenario*. Based on the delays, the delay management problem is solved by using the following heuristic. The method assumes the delays not to be known in advance and applies a rule-of-thumb where all wait-depart decisions are taken according to the following rule. If the feeder train has at most 5 minutes delay, the transfer is maintained, otherwise the transfer is not held (see [Sch10]). As mentioned in Section 2, this method (fixed waiting rule) is similar to the delay management strategy used by railway companies in practice (see [KS11]). For each line planning model, the last step is computed for 100 different delay scenarios in order to factor out the randomness of the delays.

Routing of Passengers under Delays (RD). The passengers are, similar to previous routing steps, distributed among the different departure times. Note that only such departure times are considered for which a path in the time expanded timetable exists. The routing is then computed as a shortest path where the activity weights are given by the real durations from the timetable under delays.

Evaluation Parameters The results are summarized in Table 2. The row values have the following meaning.

Row 1 Costs of line concept
Row 2 Number of direct travelers estimated at line concept level
Row 3 Number of lines used by line concept
Row 4 Average length of line used by line concept
Row 5 Average frequency per line used by line concept
Row 6 Number of passenger weighted transfers on a shortest path estimated at line concept level (RLC)
Row 7 Average buffer of transfer activities
Row 8 Passenger weighted traveling time after routing at timetabling level (RT); the duration of each activity is weighted by the passengers on the activity and the sum over all activities is computed.
Row 9 Passenger weighted buffer time after routing at timetabling level (RT); for each activity, the difference between the duration and the lower bound is weighted by the number of passengers on the activity and then summarized over all activities.
Row 10 Passenger weighted delay after routing at delay management level (RD); the difference between the duration of the travel under the planned and under the actual timetable is summarized for all passengers.

Note that these parameters are evaluated after different levels of the planning process. The level is indicated by the column "Level" in Table 2, where LP stands for line planning, TT for timetabling and DM for delay management.

Row	Level	Parameter	(**LPT**)	(**Direct**)	(**Cost**)	(**Game**)
1	LP1	costs	31841	28896	**15261**	15288
2	LP2	direct travelers	147858	**156851**	120739	121881
3	LP3	num lines	80	83	42	41
4	LP4	average line length	367028	365463	377323	387666
5	LP5	average frequency	1.21	1.00	1.00	1.00
6	LP6	transfers on shortest paths	370113	**368959**	445519	445117
7	TT1	average buffer transfers	25	26	22	21
8	TT2	passenger weighted traveling time	60952147	61083726	62490414	62273404
9	TT3	passenger weighted buffer	1744743	1837135	3042042	2983899
10	DM1	passenger weighted delay	**27057739.6**	27067834.7	53261600.3	51801233.7

Table 2: Results of testing line planning models

Results In all evaluation parameters we have strong similarity for the models designed from passenger perspective, i.e., (**Direct**) and (**LPT**), and the ones from operators perspective, i.e., (**Cost**) and (**Game**). At the stage of line planning this applies to the costs of the line concept (LP1), the number of direct travelers (LP2), the number of lines (LP3) and the number of estimated transfers on shortest paths (LP6). This indicates that the trade-off between company and passenger related interests is rather contradictory.

The evaluation parameter giving the transfers on shortest paths (LP6) attains the smallest value for the model (**Direct**). The value (LP4) is not the objective value of (**LPT**). The model (**LPT**) minimizes the transfers for each OD-pair on preferable paths and not only on the shortest path. The routing at the stage of a line concept, which underlies this parameter, routes all passengers on their shortest path with respect to the physical distance.

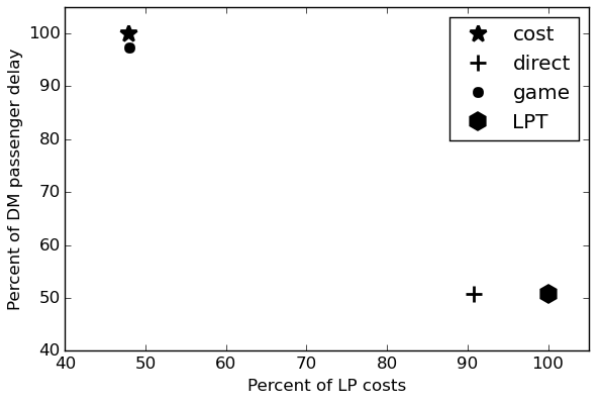


Figure 6: Trade-off between costs and delay management quality measured as percentage with respect to the maximum of the corresponding values.

There are various studies on the trade-off between a good timetable and a good delay manage-

ment (see [FSZ09a, CT12b]). Since we want to focus on the effect of the line planning model on the delay management, the timetables are computed to be of similar quality in terms of the passenger traveling time (TT2). The timetables are equalized in this measurement since it reflects the passenger perspective. Note that the models (**Cost**) and (**Game**) have almost twice as much passenger weighted buffer (TT3) compared to the other models. The reason is that the passenger oriented models provide a basis for timetables in which most of the passengers choose paths for which the traveling time is close to the shortest possible traveling time. Additionally, there are only few passenger transfers in the timetables provided by passenger related line planning models. Thus, in the objective function the traveling time on the transfers do not appear as important as the traveling time on the driving and waiting activities of the trains. It means that the buffer on passenger transfers is rather high (TT1).

In contrast, on basis of (**Cost**) and (**Game**) a timetable is provided in which many passengers choose paths along activities where a large differences between the durations and their lower bounds are. In particular, many passengers transfer which means that the traveling times of passengers on transfers appear as very important in the objective function. Thus, mainly the buffer on transfer activities is minimized in the timetabling step (TT1).

In principle, the large amount of buffer (TT3) of (**Cost**) and (**Game**) is good in terms of delay resistance since this can be used to buffer against delays. Still, in the following delay management step the contrary is indicated. Timetables based on line planning models representing the operators perspective are not as delay resistant as the ones based on the other line planning models (DM1). Indeed, (**Cost**) and (**Game**) turn out to have by far the worst delay management properties. And the reason for this is the buffer on the transfers. In the passenger oriented models the average buffer on the transfers is significantly higher than in the models designed from operators perspective. It means that delays can be buffered better. In comparison, the models (**Cost**) and (**Game**) provide only half the amount of lines (LP3) with similar average lengths (LP4) and similar average frequencies (LP5). Hence, a if a connection is missed, which is more likely to happen with low buffer times, passengers have to endure relatively long waiting times, i.e., delays. Comparing (**Direct**) and (**LPT**) we see a similar quality in terms of the delay resistance.

The relation between the most important parameters, the costs and the quality of the delay management, are depicted in Figure 6. It indicates that the costs of the line concepts (LP1) are by far better for the models designed from operators perspective. On the other hand, the resulting passenger weighted delays parameter (DM1) is by far smaller for the (**Direct**) and (**LPT**) in relation to the other models. Comparing the passenger oriented models, from the figure no distinction between the direct travelers model and (**LPT**) can be made.

We hence conclude from this study that taking a close look on passenger paths and their transfers at the stage of line planning highly improves the delay resistance.

7 Discussion and Outlook

From a modeling point of view this work extends the idea of [BKZ97] by using a set of preferable paths. This can be seen as a generalization of [BGP07, SS06a] in which all paths can be used by passengers. Similar to [BGP07] we show that the problem is suitable for column generation and provide computational results supporting this claim. In comparison to these previous works we also show that much larger instances can be solved with this approach. Finally, the principle aim of this work is the applicability of the proposed line planning model as a basis for delay resistant public transportation systems. Similar to [GSS13], the software package LinTim is used to compute solutions for the consecutive planning phases of public transportation. In addition to [GSS13], we compared the delay resistance of the new model to previously existing ones and showed that the new model is both the most expensive and the most delay resistant one. This contributes to the discussion of designing the lines towards user-friendly aspects.

Also methods specifically engineered for (**LPT**) can be developed computing an integer solution on basis of a fractional solution. Until now only the Xpress Optimizer internal branch and bound procedure is applied. This could give the ability to prove whether the resulting solution is feasible and also the quality of the solution could be investigated.

One element neglected in this study is the consideration of infrastructure capacity. More work has to be done in order to study the additional difficulty of headways resolving infrastructure capacity problems. They have a big influence on the delay propagation as well.

The results of this work and in particular the computation method can also be applied to the more general traveling time model (see [SS06a]). Similarly to this work, preferable path based passenger flows can be integrated in the line planning model. Then, applying column generation might also solve this problem.

Finally, this model gives an idea for future research activities. We recognize that the decision made in the line planning has a great effect on subsequent planning problems. Hence, when studying the problems of public transportation planning it should be kept in mind. And also, it should be investigated further into the direction of clarifying the relation and dependencies between different planning stages.

References

[APTK15] Emma Andersson, Anders Peterson, and Johanna Törnquist Krasemann. Improved railway timetable robustness for reduced traffic delays–a milp approach. In *6th International Conference on Railway Operations Modelling and Analysis-RailTokyo*, 2015.

[BGP07] Ralf Borndörfer, Martin Grötschel, and Marc E Pfetsch. A column-generation approach to line planning in public transport. *Transportation Science*, 41(1):123–132, 2007.

[BGP08] Ralf Borndörfer, Martin Grötschel, and Marc E Pfetsch. Models for line planning in public transport. In *Computer-aided systems in public transport*, pages 363–378. Springer, 2008.

[BJN+98] Cynthia Barnhart, Ellis L Johnson, George L Nemhauser, Martin WP Savelsbergh, and Pamela H Vance. Branch-and-price: Column generation for solving huge integer programs. *Operations research*, 46(3):316–329, 1998.

[BKZ97] M.R. Bussieck, P. Kreuzer, and U.T. Zimmermann. Optimal lines for railway systems. *European Journal of Operational Research*, 96(1):54 – 63, 1997.

[BS04] Dimitris Bertsimas and Melvyn Sim. The price of robustness. *Operations research*, 52(1):35–53, 2004.

[BW05] Dimitris Bertsimas and Robert Weismantel. *Optimization over integers*, volume 13. Dynamic Ideas Belmont, 2005.

[BWZ97] M. R. Bussieck, T. Winter, and U.T. Zimmermann. Discrete optimization in public rail transport. *Mathematical programming*, 79(1-3):415–444, 1997.

[CCT10] Valentina Cacchiani, Alberto Caprara, and Paolo Toth. Non-cyclic train timetabling and comparability graphs. *Operations Research Letters*, 38(3):179–184, 2010.

[CFT02] A. Caprara, M. Fischetti, and P. Toth. Modeling and solving the train timetabling problem. *Operations research*, 50(5):851–861, 2002.

[CHK+14] Valentina Cacchiani, Dennis Huisman, Martin Kidd, Leo Kroon, Paolo Toth, Lucas Veelenturf, and Joris Wagenaar. An overview of recovery models and algorithms for real-time railway rescheduling. *Transportation Research Part B: Methodological*, 63:15–37, 2014.

[CT12a] V. Cacchiani and P. Toth. Nominal and robust train timetabling problems. *European Journal of Operational Research*, 219(3):727–737, 2012.

[CT12b] Valentina Cacchiani and Paolo Toth. Nominal and robust train timetabling problems. *European Journal of Operational Research*, 219(3):727 – 737, 2012.

[CvDZ98] M.T. Claessens, N.M. van Dijk, and P.J. Zwaneveld. Cost optimal allocation of rail passenger lines. *European Journal of Operational Research*, 110(3):474 – 489, 1998.

[CW86] A. Ceder and N.H.M. Wilson. Bus network design. *Transportation Research Part B: Methodological*, 20(4):331 – 344, 1986.

[DH03] G. Desaulniers and M. Hickman. Public transit. *Transportation, handbooks in operations research and management science*, pages 69–127, 2003.

[DHSS12] Twan Dollevoet, Dennis Huisman, Marie Schmidt, and Anita Schöbel. Delay management with rerouting of passengers. *Transportation Science*, 46(1):74–89, 2012.

[Die78] H. Dienst. *Linienplanung im spurgeführten Personenverkehr mit Hilfe eines heuristischen Verfahrens*. PhD thesis, Technische Universität Braunschweig, 1978. in German.

[DL05] Jacques Desrosiers and Marco E Lübbecke. *A primer in column generation*. Springer, 2005.

[DSCV13] Thijs Dewilde, Peter Sels, Dirk Cattrysse, and Pieter Vansteenwegen. Robust railway station planning: An interaction between routing, timetabling and platforming. *Journal of Rail Transport Planning & Management*, 3(3):68–77, 2013.

[FSZ09a] M. Fischetti, D. Salvagnin, and A. Zanette. Fast approaches to improve the robustness of a railway timetable. *Transportation Science*, 43(3):321–335, 2009.

[FSZ09b] Matteo Fischetti, Domenico Salvagnin, and Arrigo Zanette. Fast approaches to improve the robustness of a railway timetable. *Transportation Science*, 43(3):321–335, 2009.

[GHS] M. Goerigk, J. Harbering, and A. Schöbel. LinTim - Integrated Optimization in Public Transportation. Homepage. see http://lintim.math.uni-goettingen.de/.

[GHS15] P. Gattermann, J. Harbering, and A. Schöbel. Generation of line pools. In *Conference on Advanced Systems in Public Transport*, 2015.

[Gov98] R.M.P. Goverde. Max-plus algebra approach to railway timetable design. In *The 1998 6 th International Conference on Computer Aided Design, Manufacture and Operation in the Railway and Other Advanced Mass Transit Systems*, pages 339–350, 1998.

[GS13a] M. Goerigk and A. Schöbel. Improving the modulo simplex algorithm for large-scale periodic timetabling. *Computers and Operations Research*, 40(5):1363–1370, 2013. 10.1016/j.cor.2012.08.018.

[GS13b] Marc Goerigk and Anita Schöbel. Improving the modulo simplex algorithm for large-scale periodic timetabling. *Computers & Operations Research*, 40(5):1363–1370, 2013.

[GS14] Marc Goerigk and Anita Schöbel. Recovery-to-optimality: A new two-stage approach to robustness with an application to aperiodic timetabling. *Computers & Operations Research*, 52:1–15, 2014.

[GSS13] Marc Goerigk, Michael Schachtebeck, and Anita Schöbel. Evaluating line concepts using travel times and robustness. *Public Transport*, 5(3):267–284, 2013.

[JMS+10] Cédric Joncour, Sophie Michel, Ruslan Sadykov, Dmitry Sverdlov, and François Vanderbeck. Column generation based primal heuristics. *Electronic Notes in Discrete Mathematics*, 36:695–702, 2010.

[KK15] F. Kirchhoff and M. Kolonko. Modelling delay propagation in railway networks using closed family of distributions. *Technical Report*, 2015.

[KS11] Natalia Kliewer and Leena Suhl. A note on the online nature of the railway delay management problem. *Networks*, 57(1):28–37, 2011.

[Lie07] C. Liebchen. *Periodic Timetable Optimization in Public Transport*. Springer, 2007.

[LLER11] R. M. Lusby, J. Larsen, M. Ehrgott, and D. Ryan. Railway track allocation: models and methods. *OR spectrum*, 33(4):843–883, 2011.

[LM07] C. Liebchen and R.H. Möhring. The modeling power of the periodic event scheduling problem: railway timetables – and beyond. In *Algorithmic methods for railway optimization*, pages 3–40. Springer, 2007.

[MBP15] D. Makovsek, V. Benezech, and S. Perkins. Efficiency in railway operations and infrastructure management. 2015.

[Nac98] K. Nachtigall. Periodic network optimization and fixed interval timetables. *Deutsches Zentrum für Luft–und Raumfahrt, Institut für Flugführung, Braunschweig*, 1998.

[Opt07] Dash Optimization. Xpress-optimizer reference manual. *Dash Optimization Ltd., Englewood Cliffs, NJ*, 2007.

[Sch01] A. Schöbel. A model for the delay management problem based on mixed-integer-programming. *Electronic Notes in Theoretical Computer Science*, 50(1):1 – 10, 2001.

[Sch10] M. Schachtebeck. *Delay Management in Public Transportation: Capacities, Robustness, and Integration*. PhD thesis, Universität Göttingen, 2010.

[Sch12] A. Schöbel. Line planning in public transportation: models and methods. *OR spectrum*, 34(3):491–510, 2012.

[Sch14] M. Schmidt. *Integrating routing decisions in network problems*. Springer, 2014.

[SG13] M. Siebert and M. Goerigk. An experimental comparison of periodic timetabling models. *Computers & Operations Research*, 40(10):2251–2259, 2013.

[SS06a] A. Schöbel and S. Scholl. Line planning with minimal travel time. In *5th Workshop on Algorithmic Methods and Models for Optimization of Railways*, number 06901 in Dagstuhl Seminar Proceedings, 2006.

[SS06b] A. Schöbel and S. Schwarze. A game-theoretic approach to line planning. In *6th workshop on algorithmic methods and models for optimization of railways*, number 06002 in Dagstuhl Seminar proceedings, 2006.

[SS10a] Michael Schachtebeck and Anita Schöbel. To wait or not to wait-and who goes first? delay management with priority decisions. *Transportation Science*, 44(3):307–321, 2010.

[SS10b] M. Schmidt and A. Schöbel. The complexity of integrating routing decisions in public transportation models. In *Proceedings of ATMOS10*, volume 14. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2010.

[SS13] A. Schöbel and S. Schwarze. Finding delay-resistant line concepts using a game-theoretic approach. *Netnomics*, 14(3), 2013.

[Yen71] J.Y. Yen. Finding the k shortest loopless paths in a network. *management Science*, 17(11):712–716, 1971.

Addendum D

J. Harbering, A. Ranade and M. Schmidt
Single Track Train Scheduling
Journal of Scheduling

Single Track Train Scheduling

Jonas Harbering (jo.harbering@math.uni-goettingen.de)
Georg-August-University of Göttingen

Abhiram Ranade (ranade@cse.iitb.ac.in)
Indian Institute of Technology

Marie Schmidt (schmidt2@rsm.nl)
Erasmus University Rotterdam

Oliver Sinnen (o.sinnen@auckland.ac.nz)
University of Auckland

December 11, 2015

In this work we consider the Single Track Train Scheduling Problem. The problem consists in scheduling a set of trains from opposite sides along a single track. The track passes intermediate stations and the trains are only allowed to pass each other at those stations. This problem has a close relation to minimizing the makespan in a job shop scheduling problem with two counter routes and no preemption. We develop a lower bound on the objective value of the train scheduling problem which provides us with an easy solution method in some special cases. The contrast in complexity to the analogous job shop scheduling problem is highlighted. Additionally, we prove the pseudo-polynomial solvability for a more general setting of the train scheduling problem.

***Keywords**: Complexity Analysis; Train Timetabling; Machine Scheduling; Counter Routes*

1 Introduction

In this paper we consider a scheduling problem which is motivated by a railway application: the scheduling of trains on a single bi-directional track. This problem occurs in passenger transportation in rural areas or when scheduling freight trains, as outlined in (Kraay et al., 1991) and references therein.

In its basic version, the *Single-Track-Train-Scheduling Problem (STTS)* reads as follows: we are given a single track, running from left to right, which has to be passed by P^l trains from the left and P^r trains from the right in the least time possible. The track is divided into several block sections, each block can be occupied by only one train at the same time. However, between

the blocks we have stations which have unlimited capacity. Here trains can wait in order to let trains from the opposite direction pass.

The translation to common machine scheduling terminology is quite straightforward: trains correspond to jobs, blocks correspond to machines, the block traversal time corresponds to processing time on a machine, and our objective is to minimize the makespan.

When translating the (STTS) to a machine scheduling problem, we obtain a special case of job-shop scheduling, since we have two subsets of jobs corresponding to trains coming from the left and trains coming from the right, which pass the blocks/machines in reverse order. This is called *job shop scheduling with counter-routes in the machine scheduling context. Furthermore, we have the requirement that a train ride on a block cannot be interrupted, this is referred to as* no preemption *in machine scheduling.* The (STTS) can hence be interpreted as a *job shop scheduling problem with two counter routes and no preemption* (abbreviated as $F^{\pm}||C_{\max}$ following common scheduling notation, see (Chen et al., 1998)), under the additional assumption that the processing time of a job on a machine is the same for all jobs. This relation is detailed in Section 2.2.

While $F^{\pm}||C_{\max}$ is strongly NP-hard for three machines already (Chen et al., 1998), the assumption that processing times depend only on the machines (or, to say it in the words of the train scheduling example: that block traversing times are the same for all trains) makes the problem considerably easier. In fact, we are going to show that for any fixed number of blocks, the problem can be solved polynomially in the number of trains and the maximal block length by dynamic programming.

For the case of three blocks and equal train numbers from both sides, as well as for some other special cases, we are even able to prove a closed-form expression for the minimal makespan.

The remainder of the paper is structured as follows. In Section 2 we give a short literature overview on relevant contributions in machine scheduling including some related complexity results and in single-track train scheduling. In Section 3 we formally introduce the problem. In Section 4 we derive a lower bound on the solution value. Using this bound, in Section 5 we discuss special cases where the lower bound can be reached. For the remaining cases we develop a dynamic programming approach in Section 6 and briefly discuss how constraints like limited station capacities and waiting times at stations can be included.

This paper is an extended version of the proceedings paper (Harbering et al., 2015) at the 7th Multidisciplinary International Conference on Scheduling: Theory and Applications (MISTA). Additional results in comparison to (Harbering et al., 2015) are the upper bound in Lemma 3 and the weakly polynomial complexity in Theorem 5.5.

2 Related Problems

2.1 Train Scheduling

The general problem of scheduling or timetabling has many different facets. Since problems arising in train scheduling *on networks* are often of quite different nature, we concentrate on *single-track* train scheduling in this overview.

The problem of scheduling trains on a single track has attracted attention very early already. One of the first to lay a ground for such research is (Frank, 1966). There, two way traffic systems, similar to the ones considered here, are analyzed. The main difference to our work is that they aim at determining the minimal number of trains that serve a certain train system ensuring periodic schedules. Subsequently, a model allowing for different train speeds and including delays is considered in (Petersen, 1974). There, train systems are analyzed in order to compute the average traveling time per train, depending on the number of trains with different priorities using the same track sequence. Also in (Higgins et al., 1996), delays in train scheduling are considered. The authors aim at a tool which is suitable both for realtime decision support and for timetable evaluation. They also develop a non-linear mixed integer model to minimize the average weighted travel time which takes into account several different timetabling aspects. Finally, they solve the developed model with a branch-and-bound approach. Similarily, (Janić, 1984) develop a detailed model for scheduling trains on a single track. In their model, the computation of the line capacity is reduced to computing the capacity only on a bottleneck segment. For this bottleneck segment they are able to determine the main relations between input parameters and the capacity of the line. In (Zhou and Zhong, 2007) a mixed integer linear programming model is stated in which trains may start at trains may only pass each other at stations. Different headways are considered for consecutive trains on tracks and at stations. In order to minimize the total travelling time, a branch-and-bound procedure is proposed. Using a similar model to (Zhou and Zhong, 2007), (Castillo et al., 2009) propose a mixed integer linear model with varying departure and traveling times. Additionally, this model takes into account that headways between trains in the same direction are possibly shorter than headways for trains in different directions. They aim at minimizing the total arrival times of all trains at all stations. The model is then solved by a heuristic. Finally, (Rahman, 2013) considers a very similar model (to (Castillo et al., 2009; Zhou and Zhong, 2007)) with the objective of minimizing the arrival time of the last train at its destination, i.e. the makespan. Most constraints are adapted from (Zhou and Zhong, 2007) while some problem specific constrains are added in order to decrease computation time. The proposed model is then solved by a heuristic approach. A slightly different problem is discussed in (Brucker et al., 2005). Here, two trains in opposing directions are scheduled on a two-track segment. Now a part of one of the tracks fails. The question to be answered is how can trains be

scheduled on the same track segment in opposing direction without deviating too much from the previous schedule.

See (Rahman, 2013) for a broad overview on scheduling on a single bidirectional line and (Cacchiani and Toth, 2012; Cordeau et al., 1998) for more general train scheduling surveys.

The models in all of the discussed works model real-world train scheduling constraints in varying degree of detail. Our simplified train scheduling problem could, e.g., be considered to be a special case of the models described in (Cacchiani and Toth, 2012; Castillo et al., 2009; Rahman, 2013; Zhou and Zhong, 2007) and the solution methods described there for more general problems could also be applied to solve our special case. However, to the best of our knowledge, there is so far no complexity analysis for such problems. Hence, the possibility of solving these problems exactly in polynomial time, based, e.g., on combinatorial algorithms or linear programming, has not been ruled out yet. With this work, we aim to lay a ground for filling this gap.

Train scheduling is closely related to machine scheduling. In fact, even more general timetabling problems can be modeled using disjunctive graphs, which are also frequently used to model machine scheduling problems. See (Balas, 1969) for an introduction to this modeling approach and (Burdett and Kozan, 2010) for an application of it to timetabling.

In the next section, we detail how the (**STTS**) can be interpreted in a machine scheduling context and how complexity results from machine scheduling transfer to our problem.

2.2 Relation to Machine Scheduling

The problem (**STTS**) can be interpreted as a machine scheduling problem as follows: Let a set of machines $M_1, \ldots, M_n$ and a set of jobs $J_1^l, \ldots, J_{P^l}^l, J_1^r, \ldots, J_{P^r}^r$ be given. Then the aim is to feasibly schedule the jobs $J_1^l, \ldots, J_{P^l}^l$ on the sequence of machines $M_1 - \cdots - M_n$ and the jobs $J_1^r, \ldots, J_{P^r}^r$ on the sequence $M_n - \cdots - M_1$ while minimizing the makespan. Note that such a sequence of machines is called a *route* in machine scheduling.

The following conditions must hold for a schedule to be feasible: Jobs can only be processed by one machine at a time, machines can only process one job at a time and once the processing of a job on a machine has started, the job can not be interrupted. This problem is called the *job shop problem with two counter routes and no preemption* ($F^{\pm}//C_{max}$). This abbreviation corresponds to the usual three field representation which is used for classifying machine scheduling problems. It means that the problem is a flow shop problem (F) with two counter routes ($\pm$) and the objective is to minimize the makespan (C_{max}). See (Chen et al., 1998) for an extensive analysis of flow and job shop problems, including this problem.

In our problem, the trains are represented by the jobs and the blocks are represented by the machines. An important difference between (STTS) and $F^{\pm}//C_{max}$ is that in $F^{\pm}//C_{max}$ the

processing times of different jobs on a machine can differ, while in (STTS) we assume all trains i to have the same traversal time p_i on each block. Hence, in the above-mentioned classification scheme for scheduling problems, (STTS) corresponds to the problem $F^{\pm}/p_{ij} = p_i/C_{max}$. Hereby, $p_{ij} = p_i$ means that the processing time p_{ij} of job $j \in \{J_1^l, \ldots, J_{Pl}^l, J_1^r, \ldots, J_{Pr}^r\}$ on machine $i \in \{M_1, \ldots, M_n\}$ is independent of the job. This restriction is well-studied in job-shop scheduling ((Gonzalez, 1982; Hromkovič et al., 2007; Lenstra and Kan, 1979)) but has to the extent of our knowledge not been considered in conjunction with counter routes.

The complexity of $F^{\pm}//C_{max}$ is well researched in the literature: For $n = 2$ the job shop problem with two counter routes and no preemption can be solved in polynomial time.

E.g., (Jackson, 1956) give an $\mathcal{O}\left((P^l + P^r)log(P^l + P^r)\right)$ algorithm for a problem equivalent to $F^{\pm}n//C_{max}$. We conclude that the problem (**STTS**) can be solved in polynomial time for $n = 2$ and that this result would even hold if every train block combination had a different traversal time.

Still, already for $n = 3$ machines, job shop scheduling with two counter routes is NP-hard if processing times on the machines may differ for different jobs. This result immediately follows from the NP-hardness of flow shop scheduling on three machines with only one route (Garey et al., 1976). The result even holds for the case that the number of jobs from both sides are equal, i.e. if $P^l = P^r$, since the special case where all jobs from the right have 0 processing times is again equivalent to flow shop scheduling.

However, this complexity result does not carry over to the (STTS) where the block traversal times are equal for all jobs. Hence, the question of whether (STTS) can be solved in polynomial time or not is still open for the case of three or more blocks.

In (Dushin, 1988) an upper bound on the objective value for job shop scheduling with two routes (not necessarily counter routes) on n machines is proven. Also (Babushkin et al., 1977) develop an upper bound for the counter routes problem on n machines. In (Dushin, 1988) an earlier publication ((Babushkin et al., 1974)) on this topic is mentioned, however, we were not able to find this paper. See also the scheduling overview given by (Chen et al., 1998; Sevast'janov, 1994) for a collection of results on the $F^{\pm}//C_{max}$ problem.

3 Model

In this work we investigate the complexity of and approaches for solving the single track train scheduling problem. This problem is described as follows.

Let a linear graph $G = (V, B)$ with stations $V = \{s_1, \ldots, s_{n+1}\}$ and undirected edges, (called *blocks* in the remainder of this paper) $B = \{b_1 = (s_1, s_2), \ldots, b_n = (s_n, s_{n+1})\}$ be given. In order to facilitate the problem description, we imagine that the tracks are going from left to right.

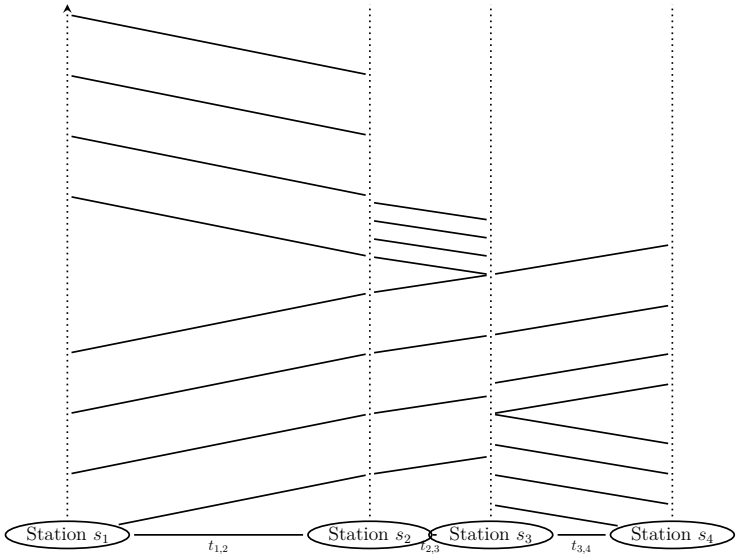


Figure 1: Example for a space time diagram with $P^l = P^r = 4$ and $n = 4$

We define station s_1 to be the leftmost station and station s_{n+1} to be the rightmost respectively.

The traversal times of the blocks are given as $t_{i,i+1}$ for block $b_i = (s_i, s_{i+1})$ for all $i = 1, \ldots, n$. Waiting times at stations are neglected, i.e., in our model a train can pass a station without stopping. The number of trains traversing the graph from s_1 to s_{n+1} (or from left to right respectively) is specified by P^l and the number of trains traversing G in opposite direction is given as P^r. At any point in time there can at most be one train on each block. The capacity of the stations is not restricted. It means that any number of trains may be stored at any station. The described setting of course oversimplifies reality. However, the intent of this paper is to investigate to which extent the (STTS) is solvable in this very simple setting. Possible extensions and ways to make the model more realistic are discussed in Section 7.

With the above notation we can now formally state the problem of *Single Track Train Scheduling* (**STTS**).

(**STTS**)
Let the graph $G = (V, B)$ with traveling times, station and block capacities be given as above. Then the aim is to minimize the makespan for the traversal of G for P^l trains from left to right and P^r trains from right to left.

In order to illustrate scheduling strategies, we make use of space time diagrams. In such diagrams, the stops of the linear network are denoted on the horizontal axis, the time is given by the vertical axis. Lines in the diagram represent the traversal of trains. Figure 1 depicts such a diagram.

For explanatory purpose we introduce two terms. Suppose a train is at a certain point of the linear network, then we call the remainder of the network, which the train still has to traverse, its *remaining part* of the network. The time-wise distance between two trains (usually on a specific block) is called *headway*.

In the following we discuss how this problem is related to machine scheduling and to what extent complexity results from scheduling theory carry over to (STTS).

4 Bounds

In this section we discuss a lower bound on the objective value of the optimal solution for an instance I of (STTS). This value will help us to prove optimality of scheduling strategies for subsequent instances of (STTS).

The lower bound is given by

$$T^{lo}(I) = \max_{i=1,\dots,n} \left\{ \left(P^l + P^r\right) t_{i,i+1} + 2 \min \left\{ \sum_{j=1}^{i-1} t_{j,j+1}, \sum_{j=i+1}^{n} t_{j,j+1} \right\} \right\} \tag{1}$$

Let the block for which the maximum is attained in (1) be called the *bottleneck block*. Later on we will see that in many cases, a good scheduling strategy on the bottleneck block guarantees that the lower bound can be obtained as the makespan.

Lemma 4.1. *$T^{lo}(I)$ is a lower bound on the objective value of an instance I of (**STTS**).*

Proof. Let us first consider one arbitrary block $b_{i,i+1}$. All trains must pass this block which makes up $(P^l + P^r)t_{i,i+1}$ of time. Additionally, the first train that passes the block needs some time to arrive at the block.

The minimum time for a train to arrive at $b_{i,i+1}$ is given by the minimum of the distances of the parts left and right of the block, i.e.

$$\min \left\{ \sum_{j=1}^{i-1} t_{j,j+1}, \sum_{j=i+1}^{n} t_{j,j+1} \right\}$$

Subsequently, all trains pass that block. Still the last train has to reach its destination. The time to reach the destination is again bounded below by the minimum of the distances of the parts left and right of the block, i.e.

$$\min \left\{ \sum_{j=1}^{i-1} t_{j,j+1}, \sum_{j=i+1}^{n} t_{j,j+1} \right\}$$

Summing up all parts we obtain for each block a value which gives a lower bound on the makespan. In particular, the maximal value of those is also a lower bound. □

This lower bound will be of help to show that particular cases can be solved in polynomial time. For the proof of the complexity of one instance of problem (STTS) we also need the upper bound on the objective value. Let therefore $i^* = \max_{i\in[2,\dots,n-1]} i$ be such that

$$\sum_{i=1}^{i=i^*-1} t_{i,i+1} \leq \frac{1}{2}\sum_{i=1}^{n} t_{i,i+1} \tag{2}$$

There are instances of (STTS) for which no such $i^* \in [2,\dots,n-1]$ exists since $t_{1,2} > \frac{1}{2}\sum_{i=1}^{n} t_{i,i+1}$. In this case we simply set $i^* = 2$. Then the upper bound $T^{up}(I)$ can be stated as

$$\begin{aligned} T^{up}(I) = &\max\left\{\sum_{i=1}^{i^*-1} t_{i,i+1} + \left(P^l-1\right)\max_{i=1,\dots,i^*-1} t_{i,i+1},\ \sum_{i=i^*+1}^{n} t_{i,i+1} + (P^r-1)\max_{i=i^*+1,\dots,n} t_{i,i+1}\right\} \\ &+ \left(P^l+P^r\right)t_{i^*,i^*+1} \\ &+ \max\left\{\sum_{i=1}^{i^*-1} t_{i,i+1} + (P^r-1)\max_{i=1,\dots,i^*-1} t_{i,i+1},\ \sum_{i=i^*+1}^{n} t_{i,i+1} + \left(P^l-1\right)\max_{i=i^*+1,\dots,n} t_{i,i+1}\right\} \end{aligned} \tag{3}$$

We now show the correctness of $T^{up}(I)$.

Lemma 4.2. *For an instance I of (STTS) $T^{up}(I)$ is an upper bound.*

Proof. Let I be an instance of (STTS) with P^l trains from the left and P^r trains from the right. First consider trains from the left. We schedule all such trains on the segments $b_1,\dots,b_{i^*-1}$ consecutively with headway $\max_{i=1,\dots,i^*-1} t_{i,i+1}$. This takes $\sum_{i=1}^{i^*-1} t_{i,i+1} + \left(P^l-1\right)\max_{i=1,\dots,i^*-1} t_{i,i+1}$ time units. Similarly, on the stretch $b_{i^*+1},\dots,b_n$ we schedule the trains from the right which amounts for $\sum_{i=i^*+1}^{n} t_{i,i+1} + (P^r-1)\max_{i=i^*+1,\dots,n} t_{i,i+1}$ time units. Then trains only start to traverse the central block b_{i^*,i^*+1} once all trains have arrived at either s_{i^*} or s_{i^*+1}. Subsequently, all trains traverse the central block summing up to $\left(P^l+P^r\right)t_{i^*,i^*+1}$ time units. Analogously to the scheduling towards the central block, we now schedule all trains from the right on the segments $b_1,\dots,b_{i^*-1}$ and all trains from the left on the segments $b_{i^*+1},\dots,b_n$ leaving the central block. This results in the upper bound $T^{up}(I)$. □

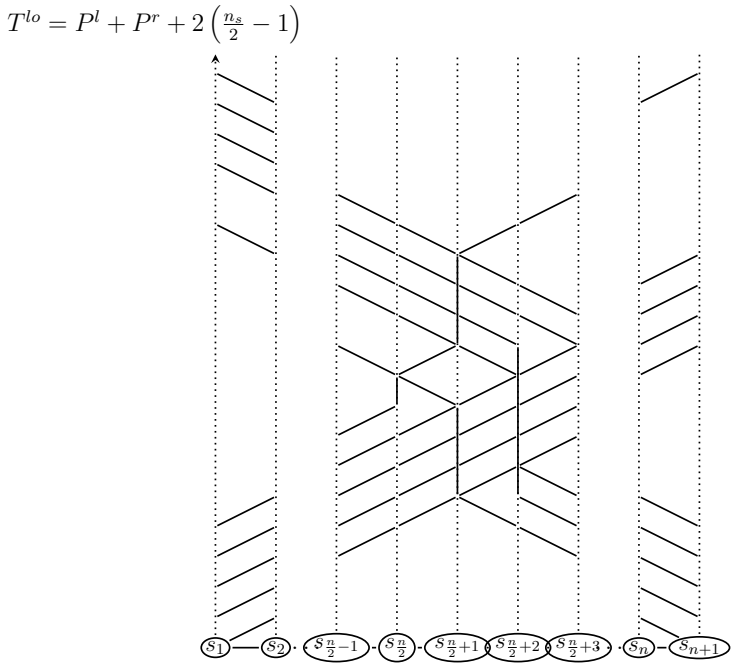


Figure 2: Strategy for unit processing time and even number of blocks

5 Special Cases

We will now direct to special cases for which we can specify scheduling strategies which allow to reach the lower bound on the makespan.

5.1 Unit processing times and capacity restrictions at stations

Assume that all traversal times on all blocks are equal, i.e. $t_{i,i+1} = 1$ for all $i = 1, \ldots, n$. Under this condition we can find optimal schedules which do not need high station capacity at intermediate stations.

Theorem 5.1. *For instances of (**STTS**) with unit processing times, the makespan of an optimal schedule is given by $T^{lo}(I)$. This even holds if the capacity at all stations is bounded by three.*

Proof. First, assume that no capacity restrictions are enforced. Two different cases will be considered. This is n is even and n is odd.

Consider the case where n is even. The following strategy provides an optimal schedule. See Figure 2 for a sketch of the schedule.

Let all trains from both sides start traversing the linear network at time 0 with a headway of one. If there is a conflict, i.e., if two trains would cross the same block at the same time, always let trains from the left precede those from the right, except for the case that the train from the

left is the P^lth - in this case give the trains from the right precedence. After all trains from the left, except the P^lth, have traversed all blocks, all trains from the right traverse the remaining part of the network. After all trains from the right have passed, the P^lth train continues to station s_{n+1}.

The last train from the left traverses the block left of the central station $s_{\frac{n}{2}+1}$ between the first and the second train from the right. Then the train waits at the central station until all trains from the right have passed and finally heads for its destination s_{n+1}. We now have to show that indeed the lower bound $T^{lo}(I)$ is obtained with this strategy.

The first train from the left and from the right reach the central station $s_{\frac{n}{2}+1}$ at the same time $\frac{n}{2}$. After that all trains from the left traverse the central part heading for station $s_n + 1$, in particular the block right of the central station. Between time $\frac{n}{2}$ and time $\frac{n}{2} + P^l - 1$, the trains $1, 2, 3, \ldots, P^l - 1$ from the left cross the block right of the central station one after another and proceed to the right. In the meantime trains from the right have started from station s_{n+1} and traversed as many blocks as possible.

At station $s_{\frac{n}{2}+1}$ there is only one train waiting. At stations $s_{\frac{n}{2}+2}, s_{\frac{n}{2}+3}, \ldots, s_n$ there are two trains waiting as long as there are trains, the remaining trains wait at station s_{n+1}.

At time $\frac{n}{2} + P^l - 2$ the first train from the right starts traversing its remaining part of the linear network. Since then there are no more trains from the right waiting at station $s_{\frac{n}{2}+1}$ and the block $b_{\frac{n}{2}+1,\frac{n}{2}+2}$ is used by the penultimate train from the left for one more time unit, the next train heading for the left has a headway of two to the first train. Hence, in this gap, the last train from the left traverses the block $b_{\frac{n}{2},\frac{n}{2}+1}$ and thus waits at station $s_{\frac{n}{2}+1}$. Finally, all trains from the right traverse their remaining part of the network with a headway of one, including those which were waiting at the station s_{n+1}. Once the last train from the right has arrived at the central station $s_{\frac{n}{2}+1}$ (at time $\frac{n}{2} + P^l + P^r - 1$) both this train and the last train from the left finish their ride to their final station. This ride takes each train $\frac{n}{2}$ time units.

If we now consider block $b_{\frac{n}{2}+1,\frac{n}{2}+2}$ we find the following sequence of trains. Until time $\frac{n}{2} - 1$ there is no train. Then the first train from the right passes this block. After that all trains from the left but the last one traverse this block. Subsequently, all trains from the right but the first pass this block. Finally, the last train passing this block is the last train from the left. Then the block is empty until the last train from the left reaches its destination.

It is easy to see that no conflicts appear on the other blocks.

In order to determine the makespan of this schedule, we consider the block $b_{\frac{n}{2}+1,\frac{n}{2}+2}$. This block is empty until time $\frac{n}{2} - 1$. Afterwards, it is occupied during $P^l + P^r$ time units: Between time $\frac{n}{2} - 1$ and time $\frac{n}{2}$, the first train from the right passes the block. After that all trains from the left but the last one traverse the block until time $\frac{n}{2} + (P^l - 1)$. Subsequently, all trains from the right but the first pass the block until time $\frac{n}{2} + (P^l - 1) + (P^r - 1)$. Finally, the last train passing this block is the last train from the left which arrives at station $\frac{n}{2} + 2$ at time

$\frac{n}{2} + P^l + P^r - 1$. Consequently, the last train from the right arrives at the leftmost station s_1 at time $\frac{n}{2} + (P^l - 1) + (P^r - 1) + \frac{n}{2} = P^l + P^r + 2\left(\frac{n}{2} - 1\right) = T^{lo}(I)$ and the last train from the left arrives at the rightmost station s_{n+1} at time $\frac{n}{2} + P^l + P^r + (\frac{n}{2} - 1) = P^l + P^r + 2\left(\frac{n}{2} - 1\right) = T^{lo}(I)$.

To see the second part of the lemma, note that at each station at each point in time there are at most two trains from the right and one from the left. Hence this schedule can be operated with a maximal station capacity of three.

Now assume that n is odd. The strategy in this case is the following. Let all trains from both side traverse the linear network with a headway of one. If there is any conflict between two trains from different directions always let trains from the left precede.

Here we find that the central block $b_{\frac{n+1}{2}, \frac{n+1}{2}+1}$ is reached by a train from each side at time $\frac{n-1}{2}$. Subsequently, all trains from the left side traverse the central block followed by all trains from the right side with headway one. Note that this is only possible since at any station from $s_{\frac{n+1}{2}+1}$ to the right there are two trains waiting. Finally, the last train has traversed the central block at time $\frac{n-1}{2} + P^l + P^r$ and it takes another $\frac{n-1}{2}$ time units until it reaches the final station s_1. Hence the makespan equals $P^l + P^r + 2\left(\frac{n-1}{2}\right)$ which equals the lower bound $T^{lo}(I)$.

To see the second part of the theorem, note that at no point in time there are more than two trains waiting at a station and a third one passing in opposing direction. Hence a maximal needed capacity of three is obtained. □

If, additionally, the number of trains from left to right and from right to left are equal, even lower station capacity is sufficient to obtain an optimal solution.

Lemma 5.2. *For instances of (STTS) with unit processing times and $P^l = P^r$, the makespan of an optimal schedule is given by $T^{lo}(I)$. This even holds if the capacity at all stations is bounded by two.*

Proof. Denote $P := P^l = P^r$. Again we separate the analysis into the cases where n is even and n is odd. First assume n is even.

The strategy consists in scheduling all trains from the left and the right at the same time with a headway of two. Then, since n is even two passing trains always meet at a station and never face a conflict on the tracks, hence, no train has to wait at intermediate stations. Consequently, the makespan is $n + 2(P - 1) = 2P + 2\left(\frac{n}{2} - 1\right) = T^{lo}(I)$.

In case n is odd, let the first train from the left start at time 0 and again let subsequent trains from the left keep a headway of two to the preceding train. Furthermore, let the first train from the right start at time 1 and let subsequent trains from the right keep a headway of two to the preceding train as well. Then, two passing trains always meet at a station and never face a conflict on the tracks, hence, no train has to wait at intermediate stations. Consequently, the

last train from the right arrives at time $1 + n + 2(P-1) = 2P + 2\left(\frac{n-1}{2}\right) = T^{lo}$, the last train from the left arrives at time $n + 2(P-1) = 2P + 2\left(\frac{n-1}{2}\right) - 1 < T^{lo}$.

Note that in both cases, no trains have to wait and that at most two trains are passing each other in stations. Hence, the capacity needed equals two. □

We conclude this section with a remark on the situation with unit capacity which also holds if block lengths are not equal. In this section, the lower bound T^{lo} cannot be reached. Since trains cannot pass each other, the optimal strategy simply consists of letting the trains from one side pass first, and then the trains from the other side. This leads to the following lemma.

Lemma 5.3. *If the station capacity is restricted to one, the makespan of an optimal schedule is given by* $2\sum_{i=1}^{n} t_{i,i+1} + (P^l + P^r - 2)\max_{i=1}^{n} t_{i,i+1}$.

5.2 Restricting the Number of Blocks

In the following we consider arbitrary track lengths, but restrict the number of blocks on the track. From the machine scheduling analysis we know that the case $n = 2$ is easily solvable. Hence, we direct to the case where $n = 3$. Take Figure 3 as an illustration of the considered case.

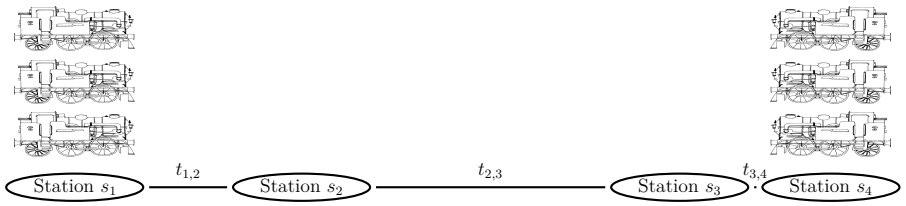


Figure 3: Example setting for $n = 3$

Here, we can find a polynomially solvable case which is given by the following theorem.

Theorem 5.4. *For instances I of (**STTS**) with three blocks and $P = P^l = P^r$, the makespan of an optimal schedule is given by $T^{lo}(I)$.*

Proof. $T^{lo}(I)$ equals the largest of the three bounds $\overbrace{2Pt_{12}}^{(1)}$, $\overbrace{2Pt_{23} + 2\min\{t_{12}, t_{34}\}}^{(2)}$, $\overbrace{2Pt_{34}}^{(3)}$.

Suppose bound (2) is largest. Then, assume wlog. that $t_{12} \geq t_{34}$. We first construct a schedule with makespan T^{lo} blockwise, then we show its validity. The first half of the schedule is composed as follows. On block b_{12}, we have P trains leaving consecutively (i.e., with headway t_{12}) to the right. Block b_{23} is initially unused until time t_{34}. After that P trains go alternately and consecutively: first to left and then to right until all trains have passed this block at time $t_{12} + 2Pt_{23}$. On block b_{34}, starting at time zero, trains go left consecutively.

The second half of the schedule is described backwards from the time $T^{lo}(I)$. On block b_{12} we have left-going trains arriving consecutively at station s_1 (i.e., they arrive at times T^{lo}, $T^{lo} - t_{12}$, $T^{lo} - 2t_{12}$,...). Block b_{23} is unused for the last duration of t_{34}. Before that we have alternately moving trains, the last rightward, the second last leftward and so on, as described above. On block b_{34} we have right-going trains arriving consecutively at times T^{lo}, $T^{lo} - t_{34}$, $T^{lo} - 2t_{34}$,... at station s_4. Note that on blocks b_{12} and b_{34} there might be unused times around the middle of the schedule.

We now argue that this is a valid schedule, i.e.,

- no two trains are in the same block at any time and
- no train is scheduled to depart from a station before it has arrived there.

We only discuss this for the first half of the schedule; the second half is analogous. The first half is $Pt_{23} + t_{34}$ long. Clearly the P movements scheduled for block b_{23} fit in this duration, after the initial inactive period of length t_{34}. Because bound (2) is largest we know that $Pt_{12}, Pt_{34} \leq Pt_{23} + t_{34}$. Thus the P rightward (leftward) movements on block b_{12} (b_{34}) can also be accommodated, and there are no two trains at the same block at any time.

Next, as described in the schedule above, the ith rightward train must leave block b_{12} at time it_{12} and enter block b_{23} at $t_{34} + (2i-1)t_{23}$. But since $t_{12} \leq t_{23} + \frac{t_{34}}{P}$, we have that

$$it_{12} \leq it_{23} + i\frac{t_{34}}{P} \leq (2i-1)t_{23} + t_{34}.$$

Thus, the train is scheduled to enter block b_{23} only after leaving block b_{12}.

Likewise, the ith leftward train is scheduled to leave block b_{34} at it_{34} and to enter block b_{23} at $(2i-2)t_{23} + t_{34}$. This is possible since it holds that

$$it_{34} = t_{34} + (i-1)t_{34} \leq t_{34} + (i-1)t_{12} \leq t_{34} + (i-1)(t_{23} + \frac{t_{34}}{P}) \leq t_{34} + (2i-2)t_{23}.$$

The last inequality in this is derived as follows. Note first that if $P = 1$ then $i = 1$ and the proof is direct. Hence consider $P > 1$. We know that $Pt_{34} \leq Pt_{23} + t_{34}$ and thus $t_{23} \geq t_{34}(P-1)/P$. This results $t_{23} \geq \frac{t_{34}}{P}$ for $P > 1$.

Next suppose bound (1) is the largest. For ease of argument, we increase t_{23} until bound (2) equals bound (1). Then, we construct the same schedule as given above. This schedule can easily be transferred to a schedule for lower values of t_{23}. In the schedule, certain time intervals are reserved for movements on block b_{23}. Of each such interval we only use a subinterval of length equal to the original value of t_{23}. This clearly does not change the makespan and maintains validity, i.e. trains use distinct time intervals on each block and are still in order in which they appear in each block.

The case of bound (3) being the largest is equivalent to the case of bound (1) being largest. □

If the number of trains from both ends are not equal, i.e. $P^l \neq P^r$, the resulting complexity is still polynomial. First note that the upper bound $T^{up}(I)$, given in (3), reduces to the following in case of $n = 3$.

$$T^{up}(I) = \max\left\{P^l t_{1,2}, P^r t_{3,4}\right\} + \left(P^l + P^r\right) t_{2,3} + \max\left\{P^l t_{3,4}, P^r t_{1,2}\right\} \tag{4}$$

With this upper bound $T^{up}(I)$ we can prove the complexity of the problem (STTS) with three blocks.

Theorem 5.5. *Instances I of (STTS) with three blocks, i.e. $n = 3$, can be solved in*

$$\mathcal{O}\left(\left(P^l + P^r\right)^2 \left(\log^2\left(P^l + P^r\right) + \log\left(P^l + P^r\right) \log\left(\max\{t_{1,2}, t_{3,4}\}\right)\right)\right) \tag{5}$$

Proof. We first show that with a given makespan T, for which a feasible schedule exists, we are able to construct a solution to I with makespan T in polynomial time. In particular, this is true if T is the optimal makespan $T = T_{opt}$.

First we consider only the two outer blocks, i.e. b_1 and b_3. We schedule all P^l trains from left as early as possible on b_1, starting at 0 and do the same for all P^r trains from right on b_3. In order to adhere to the makespan T, the last train from the right needs to arrive at station s_1 at time T at the latest. The same is true for the last train from the left at station s_4. Hence, we schedule all P^r trains from the right on b_1 such that the last train arrives at time T, the next train at time $T - t_{1,2}$ and so forth. We do the same for all P^l trains from the left on block b_3. In other words, we have created an as early as possible schedule for all trains on their first block and an as late as possible schedule for all trains on their last block. Since on each block first all trains from one side and then all trains from the other side traverse, let us call this schedule *sorted*.

We claim that for every makespan T for which a feasible schedule exists there is also a feasible sorted schedule with makespan T. We now show that by iteratively interchanging two scheduled trains on a block the schedule becomes sorted while still preserving feasibility. The following argument is symmetrically applicable to both blocks b_1 and b_3 and thus we only show it on block b_1. Assume that the schedule S is not sorted, i.e. there is a train from the right scheduled before a train from the left. If we now interchange the traversal of the train from the right with the next one from the left we obtain a new schedule S'. Then move the train from the left to the earliest possible time and the train from the right to the latest possible time (which is T in case of the last train). This resulting schedule is still feasible since the train from the left only arrives earlier at station s_2 and the train from the right only departs later from station s_2. Also the

makespan T is neither increased nor decreased, i.e. there is no change in the objective value. By iteratively applying this argument to any train from the right traversing the first block before any train from the left, we obtain a schedule S^* which is feasible and sorted.

To schedule the trains on the central block, we cast this problem into a single machine scheduling problem with jobs having release and due dates (Simons, 1978). Each train corresponds to one job i. Its release time $r_{k,i}$ is the arrival time at station s_2 for trains from the left ($k = l$) and at station s_3 for trains from the right ($k = r$). The deadline $d_{k,i}$ is the departure time from station s_3 for trains from the left ($k = l$) (according to the above created schedule on b_3) and the departure time from s_2 for trains from the right ($k = r$). Release times and deadlines are matched in order, i.e. train 1 from left has release time $r_{l,1} = 0$ and deadline $d_{l,1} = T - P^l t_{3,4}$. Train 2 from the left has release time $r_{l,2} = t_{1,2}$ and deadline $d_{l,2} = T - (P^l - 1)t_{3,4}$ and so on. The result of (Simons, 1978) is a schedule with minimal makespan or the conclusion that no feasible schedule under such release times and deadlines exists. Hence, if the algorithm of (Simons, 1978) does not find a feasible schedule on the central block then also T is not a makespan for which a feasible schedule for (STTS) exists. Otherwise, for a makespan T, for which a feasible schedule exists, the algorithm from (Simons, 1978) finds an optimal solution for this problem in $\mathcal{O}\left(\left(P^l + P^r\right)^2 \log\left(P^l + P^r\right)\right)$.

Since the release times are as early as possible and the deadlines are as late as possible due to the schedule on the outer blocks, no other assignment of release times and deadlines could increase the likelihood that the algorithm from (Simons, 1978) finds a feasible solution to the single machine problem.

The resulting single machine schedule can be directly used as the train schedule for the central block. This completes the construction of a schedule for instance I with a makespan T. The complexity of the entire schedule construction is dominated by the scheduling on the central block, because on the outer blocks it is linear in the number of trains. Hence, it is $\mathcal{O}\left(\left(P^l + P^r\right)^2 \log\left(P^l + P^r\right)\right)$.

To find the optimal schedule for I, we conduct a binary search on the makespan T in the interval given by the lower bound (1) and upper bound (4), $T \in [T^{lo}(I), T^{up}(I)]$. The complexity of the binary search steps is determined by the difference $T^{up}(I) - T^{lo}(I)$. Note that, due to the

symmetry of the linear network, we can assume wlog. that $t_{1,2} \geq t_{3,4}$.

$$\begin{aligned}
T^{up}(I) - T^{lo}(I) =& \max\left\{P^l t_{1,2}, P^r t_{3,4}\right\} + \left(P^l + P^r\right) t_{2,3} + \max\left\{P^l t_{3,4}, P^r t_{1,2}\right\} \\
& - \max_{i=1,\ldots,3}\left\{\left(P^l + P^r\right) t_{i,i+1} + 2\min\left\{\sum_{j=1}^{i-1} t_{i,i+1}, \sum_{j=i+1}^{3} t_{i,i+1}\right\}\right\} \\
=& \max\left\{P^l t_{1,2}, P^r t_{3,4}\right\} + \left(P^l + P^r\right) t_{2,3} + \max\left\{P^l t_{3,4}, P^r t_{1,2}\right\} \\
& - \max\left\{\left(P^l + P^r\right) t_{2,3} + 2t_{3,4}, \left(P^l + P^r\right) t_{1,2}\right\} \\
=& \max\left\{P^l t_{1,2}, P^r t_{3,4}\right\} + \max\left\{P^l t_{3,4}, P^r t_{1,2}\right\} - \max\left\{2t_{3,4}, \left(P^l + P^r\right)(t_{1,2} - t_{2,3})\right\} \\
\leq& 2t_{1,2} \max\left\{P^l, P^r\right\} - \max\left\{2t_{3,4}, \left(P^l + P^r\right)(t_{1,2} - t_{2,3})\right\} \\
\leq& 2t_{1,2} \max\left\{P^l, P^r\right\}
\end{aligned}$$

The analogous analysis is obtained for $t_{3,4} \geq t_{1,2}$. Hence, the number of steps in the binary search is $\mathcal{O}\log\left(\left(\max\{t_{1,2}, t_{3,4}\}\max\{P^l, P^r\}\right)\right)$ and the resulting total complexity is

$$\begin{aligned}
& \mathcal{O}\left(\left(P^l + P^r\right)^2 \log\left(P^l + P^r\right) \log\left(\max\{t_{1,2}, t_{3,4}\}\max\{P^l, P^r\}\right)\right) \\
=& \mathcal{O}\left(\left(P^l + P^r\right)^2 \left(\log^2\left(P^l + P^r\right) + \log\left(P^l + P^r\right) \log\left(\max\{t_{1,2}, t_{3,4}\}\right)\right)\right)
\end{aligned}$$

□

Note that the algorithm is weakly polynomial due to the term $\log(\max\{t_{1,2}, t_{3,4}\})$ in the complexity expression.

5.3 Can the lower bound $T^{lo}(I)$ always be achieved?

The following example shows a case for $n = 5$ in which the lower bound $T^{lo}(I)$ can not be achieved.

Example 5.6. Assume $P^l = P^r = 2$ and $t_{1,2} = t_{5,6} = 10$, $t_{2,3} = t_{4,5} = 3$ and $t_{3,4} = 4$. Then $T^{lo}(I) = (P^l + P^r)t_{3,4} + 2(t_{1,2} + t_{2,3}) = 42$ and the bottleneck is block $b_{3,4}$. Figure 4 shows an optimal scheduling strategy. In this strategy there is an unavoidable gap of 2 time units on the bottleneck block between the first and the second train from the right. Due to this gap, the makespan of this solution is $44 = T^{lo} + 2$.

As a general observation, we conclude that in order to reach the lower bound T^{lo}, we must be able to schedule the trains on the bottleneck block(s) without leaving a gap.

Observation 5.7. If there exists a $k = 1, 2, \ldots, 2P$ such that at time $\min\left\{\sum_{j=1}^{i-1} t_{j,j+1}, \sum_{j=i+1}^{n} t_{j,j+1}\right\} + (k-1)t_{jj+1}$ less than k trains have reached the bottleneck block, there does not exist a schedule

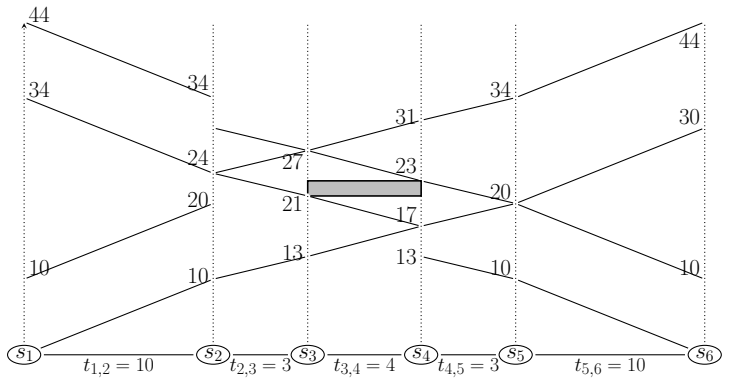


Figure 4: Sketch for scheduling strategy for Example 5.6

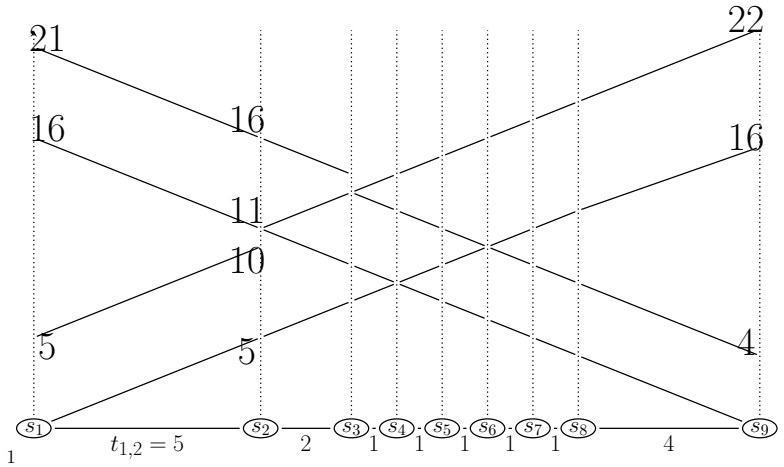


Figure 5: Lower bound T^{lo} not reached even though longest block is bottleneck

which achieves makespan T^{lo}.

Note that this can occur, even if the longest block is the bottleneck block, as can be seen in the following example.

Example 5.8.

Let $P^l = P^r = 2$ and $(t_{1,2}, t_{2,3}, \ldots, t_{8,9}) = (5, 2, 1, 1, 1, 1, 1, 4)$. Then the bottleneck block is obtained as $b_{1,2}$ with $T^{lo} = 20$. When scheduling, we see that once the second train from left has passed the bottleneck, the first train from the right has not yet reached the bottleneck. Hence the lower bound is not reached. See Figure 5 where the scheduling strategy is depicted.

6 Dynamic Programming

In this section we develop a dynamic programming formulation for the problem (STTS) with pseudo-polynomial running time.

Lemma 6.1. *If block lengths are integer, (STTS) can be solved as a shortest-path problem in an acyclic graph.*

Proof. Denote by $L := \sum_{i=1}^{n} t_{i,i+1}$ the total length of the track considered. We divide the tracks into L subblocks c_k of lengths 1 with substations s'_k at every end of a subblock. We denote by I the index set of the substations which are stations and by $I(b_j)$ the index set of the substations corresponding to block b_j, including the stations at both ends of the block. Note that each substation index i belonging to a station s_j (except for $j = 1$ and $j = n + 1$) is contained in two sets $I(b_{j-1})$ and $I(b_j)$.

States: The nodes of the graph in which we are going to formulate the shortest path problem denote the states of the problem. Each state is represented by a tuple X with entries

$$X := (x_0^l/x_0^r, x_1^l/x_1^r, \ldots, x_L^l/x_L^r),$$

also written as

$$X := \begin{array}{c|c|c|c|c|c} x_0^l & x_1^l & x_2^l & \ldots & x_{L-1}^l & x_L^l \\ \hline x_0^r & x_1^r & x_2^r & \ldots & x_{L-1}^r & x_L^r \end{array}. \tag{6}$$

Herein, x_i^l represents the number of trains from the left which have passed or reached substation s'_i already, and x_i^r represents the number of trains from the right which have passed or reached substation s'_i already.

For $n \in \mathbf{N}$ we denote $[n] := \{0, 1, 2, \ldots, n\}$.

Feasible states: Since we have P^l trains from the left and P^r trains from the right, $X^l := (x_0^l, x_1^l, \ldots, x_L^l) \in [P^l]^{L+1}$ and $X^r := (x_0^r, x_1^r, \ldots, x_L^r) \in [P^r]^{L+1}$. However, not all vectors $Y \in [P^l]^{L+1} \times [P^r]^{L+1}$ define feasible states of the problem:

- At the beginning, all trains passing from left are on the left of the track, i.e., $x_0^l = P^l$ for all feasible states X, analogously $x_L^r = P^r$ for all feasible states X.
- Since trains pass from left to right, we have $x_i^l \geq x_{i+1}^l$ for all $i = 0, \ldots, L$, analogously $x_i^r \leq x_{i+1}^r$.
- Only one train can be on a block at a time.

$$(\max_{i \in I(b_j)} x_i^l - \min_{i \in I(b_j)} x_i^l) + (\max_{i \in I(b_j)} x_i^r - \min_{i \in I(b_j)} x_i^r) \leq 1 \quad \text{for all } j = 1, \ldots, n$$

Only nodes representing feasible states will be introduced.

Transitions/arcs Two nodes $(X, \hat{X})$ are connected by a directed arc of length 1, if state $\hat{X}$ can be reached from state X in time 1. This is the case, if

- $x_i^k \leq \hat{x}_i^k$ for all $i = 0, \ldots, L$, $k \in \{l, r\}$, i.e., over time, more and more trains pass every substation of the network.
- $\sum_{i \in I(b_j)} (\hat{x}_i^l + \hat{x}_i^r) \leq \sum_{i \in I(b_j)} (x_i^l + x_i^r) + 1$ for all $j = 1, \ldots, n$, i.e., on each block there is at most one train running and it advances at most one substation.
- If for two substations s_i' and s_{i+1}', belonging to the same block b_j, $x_i^l = x_{i+1}^l + 1$, then it follows $\hat{x}_{i+1}^l = x_{i+1}^l + 1$, unless s_i' is a station. Analogously, $x_i^r = x_{i-1}^r + 1$, then $\hat{x}_{i-1}^r = x_{i-1}^r + 1$, unless s_{i+1}' is a station. This models that trains cannot stop between the stations.

These conditions ensure that the graph is acyclic.

Correspondence of solutions to paths There is a one-to-one correspondence between feasible schedules for the (STTS) and paths from node

P^l	0	0	...	0	0
0	0	0	...	0	P^r

to node

P^l	P^l	P^l	...	P^l	P^l
P^r	P^r	P^r	...	P^r	P^r

where the length of the path represents the time duration to execute the solution.

Hence, an optimal solution to (STTS) corresponds to a shortest path is the graph. □

Lemma 6.2. *If block lengths are integer, the graph described in Lemma 6.1 has* $O\left(\binom{P^l+n}{n}\binom{P^r+n}{n} \max\{t_{i,i+1}\}\right)$ *nodes and* $O\left(3^n \binom{P^l+n}{n}\binom{P^r+n}{n} \max\{t_{i,i+1}\}\right)$ *arcs.*

Proof. As stated in the proof of Lemma 6.1, the nodes of the graph correspond to feasible states X (as described in (6)). We now make an attempt on bounding the number of feasible states from above.

We denote by y_j^l the number of trains which have passed the full block b_j from left to right. y_j^r analogously denotes the number of trains which have passed the full block b_j from right to left. I.e, $y_j^l = x_{i_{\text{last}^j}}^l$ and $y_j^r = x_{i_{\text{first}^j}}^l$, where i_{first^j} and i_{last^j} denote the first and the last index in the set $I(b_j)$, respectively.

For every pair (y_j^l / y_j^r) there are $2t_{j,j+1} - 1$ feasible combinations of subindices:

$$\begin{array}{c|c|c|c|c|c} y_j^l = x^l_{i_{\text{first}^j}} & x^l_{i_{\text{first}^j}+1} & x^l_{i_{\text{first}^j}+2} & \dots & x^l_{i_{\text{last}^j}-1} & y^l_{j+1} = x^l_{i_{\text{last}^j}} \\ \hline y_j^r = x^r_{i_{\text{first}^j}} & x^r_{i_{\text{first}^j}+1} & x^r_{i_{\text{first}^j}+2} & \dots & x^r_{i_{\text{last}^j}-1} & y^r_{j+1} = x^r_{i_{\text{last}^j}} \end{array},$$

namely

$$\begin{array}{c|c|c|c|c|c} y_j^l & y_j^l - 1 & y_j^l - 1 & \dots & y_j^l - 1 & y_j^l - 1 \\ \hline y_j^r & y_j^r & y_j^r & \dots & y_j^r & y_j^r \end{array}, \quad \begin{array}{c|c|c|c|c|c} y_j^l & y_j^l & y_j^l - 1 & \dots & y_j^l - 1 & y_j^l - 1 \\ \hline y_j^r & y_j^r & y_j^r & \dots & y_j^r & y_j^r \end{array}, \quad \dots, \quad \begin{array}{c|c|c|c|c|c} y_j^l & y_j^l & y_j^l & \dots & y_j^l & y_j^l \\ \hline y_j^r & y_j^r & y_j^r & \dots & y_j^r & y_j^r, \end{array},$$

and

$$\begin{array}{c|c|c|c|c|c} y_j^l & y_j^l & y_j^l & \dots & y_j^l & y_j^l \\ \hline y_j^r - 1 & y_j^r - 1 & y_j^r - 1 & \dots & y_j^r - 1 & y_j^r \end{array}, \quad \dots, \quad \begin{array}{c|c|c|c|c|c} y_j^l & y_j^l & y_j^l & \dots & y_j^l & y_j^l \\ \hline y_j^r - 1 & y_j^r & y_j^r & \dots & y_j^r & y_j^r \end{array}, \quad \begin{array}{c|c|c|c|c|c} y_j^l & y_j^l & y_j^l & \dots & y_j^l & y_j^l \\ \hline y_j^r & y_j^r & y_j^r & \dots & y_j^r & y_j^r, \end{array},$$

(one case is listed twice here), if $y_j^l, y_j^r > 0$. If one or both values are 0, there are less.

Furthermore, there are

$$\sum_{i_n=0}^{P^l} \sum_{i_{n-1}=0}^{i_n} \sum_{i_{n-2}=0}^{i_{n-1}} \dots \sum_{i_1=0}^{i_2} 1 \cdot \sum_{i_n=0}^{P^r} \sum_{i_{n-1}=0}^{i_n} \sum_{i_{n-2}=0}^{i_{n-1}} \dots \sum_{i_1=0}^{i_2} 1$$

feasible combinations of the y-values. Each nested sum

$$\sum_{i_n=0}^{P^l} \sum_{i_{n-1}=0}^{i_n} \sum_{i_{n-2}=0}^{i_{n-1}} \dots \sum_{i_1=0}^{i_2} 1 = \binom{P^l + n}{n}$$

then gives a binomial coefficient, see (Butler and Karasik, 2010) for an explanation. This leads to $O\left(\binom{P^l+n}{n}\binom{P^r+n}{n} \max\{t_{i,i+1}\}\right)$ nodes.

To count the number of arcs, let us consider possible successors $\hat{X}$ of node X in the graph. For each block, there are at most three different possibilities: either a train is moving from left to right, a train is moving from right to left, or no train is moving at all on that block (as formalized when defining the arcs above). Since we have n blocks, each node X has hence at most 3^n successors (and most will have much less). Thus, there are at most $O\left(3^n\binom{P^l+n}{n}\binom{P^r+n}{n} \max\{t_{i,i+1}\}\right)$ arcs in the graph. □

Lemma 6.3. *The (STTS) can be solved in* $O\left(3^n\binom{P^l+n}{n}\binom{P^r+n}{n} \max\{t_{i,i+1}\}\right)$.

Proof. Since the described graph is a directed and acyclic graph, we can find a shortest path in linear time in the number of edges, see, e.g., (Cormen et al., 2001). □

Of course, for large numbers of blocks n this is impractical and we would probably be better off using an integer programming approach as described, e.g., in (Cacchiani and Toth, 2012). However, from a theoretical point of view, the following conclusion is interesting:

Corollary 6.4. For a fixed number of blocks, the (STTS) can be solved in pseudo-polynomial time, i.e., it is polynomial in P^l, P^r and $\max_j t_{j,j+1}$.

Note that the complexity for (STTS) with three blocks stated in Theorem 5.5 is considerably smaller than the one stated in Lemma 6.3. For $n = 3$ and $P^l = P^r = P$ we obtain a complexity from Theorem 5.5 of

$$\mathcal{O}\left(P^2\left(\log^2 P + \log P \log\left(\max\left\{t_{1,2}, t_{3,4}\right\}\right)\right)\right)$$

which is very much smaller than what is obtained from Lemma 6.3:

$$\mathcal{O}\left(P^6 \max_{i=1,\ldots,3}\{t_{i,i+1}\}\right)$$

Possible Extensions In the basic version of the (STTS) we assumed that stations have an unlimited number of tracks. However, limited station capacity could easily be taken into account in the dynamic programming approach. Note that for each state X of the dynamic program, the number of trains currently halting at station s_j is

$$H_j(X) := \underbrace{x^l_{i^j_{first}} - x^l_{i^j_{first}+1}}_{\text{trains from the left}} + \underbrace{x^r_{i^j_{first}} - x^r_{i^j_{first}-1}}_{\text{trains from the right}}.$$

Hence, track capacity restrictions at stations can be included by creating only the nodes which satisfy $H_j(X) \leq$ capacity of station s_j and the corresponding arcs.

We can also include simple vehicle scheduling constraints in the model. I.e., consider the situation that the trains go back and forth and traverse the track several times during a day. In this case, we would denote by P^l the number of departures from left and by P^r the number of departures from right. Introducing only nodes X which fulfill

$$x^l_2 - x^r_1 \leq \text{initial number of trains on left}$$

we would make sure that the number of trains which have left the left station never exceeds the number of trains which have arrived there by more than the initial number of trains on the left. It makes sure that there is always a train to leave when there is a scheduled departure. An analogous constraint can be added for the station on the right.

A similar approach could be taken when introducing simple 'balance' constraints. Particularly when transporting passengers, it could be considered unbalanced and unfair, if a large number

of trains from the left could pass the full track unhindered while all trains from the right had to wait. To avoid this problem, we could, similarly to the approach described above, exclude all nodes modeling states where, e.g., x_L^l and x_0^r are too different, w.r.t. to the above stated balancedness, or where the difference between x_i^l and x_i^r at some intermediate substation s_i' is too big.

Note that all extensions described so far lead to a smaller network and hence would speed-up the dynamic programming approach.

Other extensions, like minimal waiting times at stations or different train types with different speed profiles could also be included, but *more* states (that is: more nodes) would be needed. However, the general theoretic result of pseudopolynomial solvability for a fixed number of blocks would remain valid also in these cases.

More sophisticated extensions like, e.g., periodicity or routing in stations seem to be out of the scope of this approach.

7 Conclusion

In this paper we studied a basic version of the Single Track Train Scheduling Problem, which can be considered a special case of job shop scheduling with two counter routes and no preemption. Our special case, furthermore, restricts the processing time of a job on a machine to be the same for all jobs.

We were able to prove a lower bound and an upper bound on the minimum makespan of the (STTS) and identify several special cases where the lower bound is tight.

In particular, we found that, although the job shop scheduling problem with two counter routes, no preemption, three machines and equal number of jobs from both sides is strongly NP-hard, that, if we have the additional requirement that the processing times of all jobs on each machine are equal (as it is the case in the (STTS)), we can specify scheduling strategies which reach a makespan equal to the lower bound. Even more, we found that the job shop scheduling problem with two counter routes, no preemption and three machines can be solved in weakly polynomial time.

Furthermore, we showed that for any fixed number of blocks (STTS) can be solved in pseudo-polynomial time.

This result can also be generalized for some less basic versions of single track train scheduling which even have the potential to speed-up the algorithm considerably. However, the computation time of our approach increases exponentially in the number of blocks, hence the result is more of a theoretical value and most likely not applicable in real-world train scheduling where the number of blocks is typically large.

In the general cases the lower bound may help to prove optimality or give a good estimate of distance to optimality when applying heuristic solution methods.

To what extent the results in this work can be generalized and the lower bound can be improved is currently under research.

Acknowledgement

The authors were partially funded by by the European Union Seventh Framework Programme (FP7-PEOPLE-2009-IRSES) under grant number 246647 with the New Zealand Government (project OptALI). We also thank the Simulationswissenschaftliches Zentrum Clausthal-Göttingen (SWZ) for financial support.

References

A. I. Babushkin, A. A. Bashta, and I. S. Belov. Scheduling for the problem with oppositely directed routes. *Kibernetika*, 7:130–135, 1974.

A. I. Babushkin, A. A. Bashta, and I. S. Belov. Scheduling for problems of counterroutes. *Cybernetics and Systems Analysis*, 13(4):611–617, 1977.

E. Balas. Machine sequencing via disjunctive graphs: an implicit enumeration algorithm. *Operations research*, 17(6):941–957, 1969.

P. Brucker, S. Heitmann, and S. Knust. *Scheduling railway traffic at a construction site*. Springer, 2005.

R. L. Burdett and E. Kozan. A disjunctive graph model and framework for constructing new train schedules. *European Journal of Operational Research*, 200(1):85–98, 2010.

S. Butler and P. Karasik. A note on nested sums. *Journal of Integer Sequences*, 13(2):3, 2010.

V. Cacchiani and P. Toth. Nominal and robust train timetabling problems. *European Journal of Operational Research*, 219(3):727–737, 2012.

E. Castillo, I. Gallego, J. M. Ureña, and J. M. Coronado. Timetabling optimization of a single railway track line with sensitivity analysis. *TOP*, 17(2):256–287, 2009.

B. Chen, C. Potts, and G. Woeginger. A Review of Machine Scheduling: Complexity, Algorithms and Approximability. *Handbook of Combinatorial Optimization*, 1998.

J.-F. Cordeau, P. Toth, and D. Vigo. A survey of optimization models for train routing and scheduling. *Transportation Science*, 32(4):380–404, 1998.

T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to algorithms*, volume 2. MIT press Cambridge, 2001.

B. I. Dushin. An algorithm for the solution of the two-route johnson problem. *Cybernetics and Systems Analysis*, 24(3):336–343, 1988.

O. Frank. Two-way traffic on a single line of railway. *Operations Research*, 14(5):801–811, 1966.

M. R. Garey, D. S. Johnson, and R. Sethi. The Complexity of Flowshop and Jobshop Scheduling. *Mathematics of Operations Research*, 1976.

T. Gonzalez. Unit execution time shop problems. *Mathematics of Operations Research*, 7(1): 57–66, 1982.

J. Harbering, R. Ranade, and M. Schmidt. Single Track Train Timetabling. In *In proceedings of the 7th Multidisciplinary International Conference on Scheduling : Theory and Applications (MISTA 2015)*, 2015.

A. Higgins, E. Kozan, and L. Ferreira. Optimal scheduling of trains on a single line track. *Transportation Research Part B: Methodological*, 30(2):147–161, 1996.

J. Hromkovič, T. Mömke, K. Steinhöfel, and P. Widmayer. Job shop scheduling with unit length tasks: bounds and algorithms. *Algorithmic Operations Research*, 2(1), 2007.

J.R. Jackson. An extension of johnson's result on job lot scheduling. *Naval Research Logistics Quarterly*, 1956.

M. Janić. Single track line capacity model. *Transportation Planning and Technology*, 9(2): 135–151, 1984.

D. Kraay, P. T. Harker, and B. Chen. Optimal pacing of trains in freight railroads: Model formulation and solution. *Operations Research*, 39(1):pp. 82–99, 1991.

J. K. Lenstra and A. R. Kan. Computational complexity of discrete optimization problems. *Annals of Discrete Mathematics*, 4:121–140, 1979.

E. R. Petersen. Over-the-road transit time for a single track railway. *Transportation Science*, 8 (1):65–74, 1974.

S. A. A. Rahman. *Freight Train Scheduling on a Single Line Network*. PhD thesis, The University of New South Wales, 2013.

S.V. Sevast'janov. On some geometric methods in scheduling theory: a survey. *Discrete Applied Mathematics*, 55(1):59 – 82, 1994.

B. Simons. A fast algorithm for single processor scheduling. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 246–252, 1978.

X. Zhou and M. Zhong. Single-track train timetabling with guaranteed optimality: Branch-and-bound algorithms with enhanced lower bounds. *Transportation Research Part B: Methodological*, 41(3):320–341, 2007.

Addendum E

J. Manitz et. al.
Network-based Source Detection for Propagation Processes on Complex Networks with Application to Train Delays on Railway Systems
Journal of the Royal Statistical Society: Series-C

Network-based Source Detection for Propagation Processes on Complex Networks with Application to Train Delays on Railway Systems

Juliane Manitz[1], Jonas Harbering[2], Marie Schmidt[3], Thomas Kneib[1], and Anita Schöbel[2]

[1]Department of Statistics and Econometrics, Georg-August University Göttingen, Göttingen, Humboldtallee 3, 37073 Göttingen, Germany

[2]Institute for Numerical and Applied Mathematics, Georg-August University Göttingen, Göttingen, Lotzestraße 16-18, 37083 Göttingen, Germany

[3]Department of Technology and Operations Management, Rotterdam School of Management (RSM), Erasmus University Rotterdam, Rotterdam, Burgemeester Oudlaan 50, 3062 PA Rotterdam, Netherlands

The correct identification of the source of a propagation process is crucial for research questions in a wide range of research fields. Exemplarily, we consider source determination of train delays in railway systems, which mimic many-faceted diffusion patterns. We enhance a structurally quite general network-based approach for the source identification of propagation processes on networks, which requires only a minimum data basis. In extensive simulation studies on different network structures and an application to the German railway system, we examine the performance of the suggested method in comparison to a new backtracking method, which we specifically designed for delay propagation.

Keywords: Complex Network; Statistical Network Analysis; Public Transportation Network; Propagation Process; Source Detection; Train Delay

1 Introduction

Train delays can never be entirely avoided, but their impact has to be kept to a strict minimum. Between April 2012 and March 2013, it has been recorded that 20.4% of the German long-distance high-speed trains were more than five minutes delayed (on average 15 minutes, see Plöchinger and Jaschensky, 2013). In general, regional trains are less sensitive to delays, so that only 4.9% of the trains arrived late (Kuhr, 2013). However, delays are not only annoying for costumers of public transportation, but do also represent a significant economic burden for the systems operator. In this work, we consider extensive delay spreading examples which affect large parts of the German railway system, of which the vast majority is caused in one particular station.

As common as delays in *public transportation networks* (PTNs) are, as difficult is it to avoid them. A key element can be the successful identification of the delay origin from a specific delay pattern. For that, it is important to distinguish between source and propagated delays. In case of compensatory damages, legal responsibility is delegated to the causative network operator or the specific railway company. Furthermore, it is of interest to investigate if the delay cause can be dissolved or avoided. Additionally, the delay propagation process can be predicted for future time periods. With that, the timetables can be adapted starting with the following commencement of buisiness. In this context, it is surprisingly complicated to track source delays in PTNs, because of the complex composition of delays, while the recorded data is usually very inaccurate and ambiguous (Yabuki et al., 2015).

For a mathematical formalization of delay spreading, we assume a PTN with a line plan, and a pre-defined timetable. Exterior local influences such as large building sites improving the railway system or adjacent constructions, extreme weather events (e.g., lightening strikes destroying infrastructure), large regional demonstrations, and significant malfunction of a specific local system introduce disturbance in form of delays into the timetable. Those source delays are then propagated, because of dependencies between the trains due to passenger transfers or track occupation of subsequent trains. There are various possibilities for decisions which passenger transfers can be maintained and regarding the sequence of trains running along a track (*delay management strategy*). This causes a varying delay impact as well as diverse delay propagation mechanisms that lead to various different spreading configurations. Since these delay spreadings on PTNs are highly complex and of irregular patterns, they can be mathemacally well described by stochastic processes on complex networks.

However, there is few work analyzing patterns of delay spreading in PTNs (Yabuki et al., 2015). Currently, the network-theoretic analysis focuses on empirical investigation of structural properties of PTNs such as small-world characteristics. Examples are diverse: railway systems (Li and Cai, 2007; Sen et al., 2003), urban subway systems (Seaton and Hackett, 2004;

Angeloudis and Fisk, 2006), bus and tramway networks (Sienkiewicz and Holyst, 2005), entire city systems (von Ferber et al., 2009), as well as the worldwide air transportation network (Guimera et al., 2005) or the global cargo shipping (Kaluza et al., 2010). A few approaches were suggested to deduce the source of complex spreading patterns in other applications such as infectious disease epidemiology (Prakash et al., 2012; Fioriti and Chinnici, 2012; Pinto et al., 2012; Comin and da Fontoura Costa, 2011), computer science (Shah and Zaman, 2010) or communication studies (Lappas et al., 2010; Shah and Zaman, 2012; Adar and Adamic, 2005). One method utilizes maximum likelihood estimates and requires an enormous amount of data (Shah and Zaman, 2010, 2012; Pinto et al., 2012). An alternative approach reconstructs the starting point of a spreading process as the network node with high centrality within the subnetwork affected by the spreading process (Comin and da Fontoura Costa, 2011). We use an adapted version of the second method for comparison purposes. However, knowledge about the precise transmission tree is usually not given and has to be estimated.

To overcome this flaw, in this work, we use another approach towards source reconstruction of propagation processes proposed by Manitz et al. (2014), which has initially been developed to reconstruct the origin of foodborne disease outbreaks. This approach requires only a minimum data basis, assembling data about the network structure and a measure for the event magnitude observed at the network nodes. With that, we aim to find the starting point of more general propagation processes on complex networks from data about the observed event counts at the network nodes. We assume, that the most likely source of a spreading pattern can be simply estimated as the network median as defined in location theory, which takes into account the process magnitude observed at the network nodes. To obtain more robustness for noisy data, we consider not only the process event, but also the observed magnitude. Furthermore, we are able to avoid the variance estimation, which tends to be instable in particular for a low number of nodes with events in the system. Additionally to this structurally quite general approach for source detection, we suggest a recursive backtracking algorithm, that is specifically designed for propagation mechanisms of delays in PTNs.

The identification of initial delays in PTNs is a many-faceted example of source detection of general propagation processes on complex networks (Yabuki et al., 2015; Büker and Seybold, 2012). Based on a well-defined network, the application has the advantage that good models for delay propagation and the efficient simulation software such as LinTim (Goerigk et al., 2015) exist, so that various complex diffusion patterns can be mimicked. Hence, delay propagation on railway networks is a good candidate example to test whether the network-based approach from Manitz et al. (2014) can be applied for source detection problems other than the spreading of food-borne diseases. We evaluate the performance in dependency of time and source node centrality, but also analyze the robustness for different delay propagation mechanisms and PTN structures. Additionally, we assess the influence of background noise. This opens the possibility of its application to various source estimation problems in many research fields: the roots of

retards in supply chains processes (Giannakis and Louis, 2011), initial failure detection during blackouts in power grids (Crucitti et al., 2004; Albert et al., 2004), the origin of computer virus attacks in the Internet (Shah and Zaman, 2012), the source of invasive species in ecology (Stevenson et al., 2012), the beginning of rumor or misinformation in social networks (Comin and da Fontoura Costa, 2011), but also the reconstruction of the epicenter of infectious disease outbreaks (Pinto et al., 2012; Fioriti and Chinnici, 2012). Only with the knowledge of the origin of a propagation process, one is able to truly combat further spreading. Additionally, the origin is the basis for the prediction of the propagation process on complex networks.

The paper is structured as follows. In Section 2, we introduce various PTNs such as the German railway, the Athens metro and the Göttingen bus system, and compare their structural properties to others reported in the literature. In Section 3, the network-theoretic methods for source estimation are explained. This includes the general network-based source detection method and the recursive backtracking algorithm designed specifically for delays in PTNs. Source detection performance of the suggested approaches is analyzed in an extensive simulation study in Section 4. Based on different networks, we are able to mimic diverse propagation mechanisms and obtain different interesting spreading patterns. It follows the application of the methods to delay propagation data examples provided by the German railway operator *Deutsche Bahn*. Finally, we conclude our findings in Section 6.

2 Public Transportation Network Data

In Public Transportation Networks (PTNs), nodes represent line stops or stations. Two nodes are connected by a link if there is a direct connection by a scheduled line between the stops. Hence, the PTNs naturally consist of one connected component. In this work, we use delay data on four different transportation networks (see Table 1). We compare their structural characteristics with those from other PTNs reported in the literature, but also to the food shipping network, which was constructed to estimate the origin of food-borne disease outbreaks in Germany (Manitz et al., 2014).

Table 1: **Empirical network characteristics for different public transportation networks** in comparison with ranges reported in von Ferber et al. (2009, world city networks) and Sienkiewicz and Holyst (2005, Polish bus and tram systems), and the food shipping network based on the gravity model constructed to identify the origin of foodborne disease outbreaks in Germany. The *density* is the proportion of all possible links that are actually present, the *average degree* is the mean number of links connected to a node, the *average path length* is the average value of all shortest path lengths between all possible pairs of nodes in the graph, the *diameter* is the greatest shortest path length between any two nodes in the network, and *transitivity* measures the local clustering by the empirical probability for a link between two neighbors of a node (for more details see Manitz et al., 2014; Kolaczyk, 2009).

		number of nodes	number of links	density	average degree	average path length	diameter	transitivity
Application								
German railway		1049	3484	0.01	6.64	4.23	11	0.17
PTNs from LinTim								
German railway		319	446	0.01	2.80	9.40	25	0.14
Göttingen bus		257	548	0.01	4.26	13.22	35	0.12
Athens metro		51	52	0.04	2.04	9.67	29	0.00
PTNs in literature								
World city	[min]	1494	5849	0.00	2.18	6.4	27	0.022
networks	[max]	44629	52885		3.73	52	210	0.140
Polish bus	[min]	152	220	0.00	2.53	7.86		0.032
& tram	[max]	2811	3978	0.01	3.08	21.52		0.161
Food shipping network								
Germany		412	30646	0.18	148.77	2.35	5	0.64

For the application, we use data on the German railway system provided by the major operator *Deutsche Bahn* (see Section 5). This network consists of 1,049 nodes and 3,484 links, so that we observe a very low density of 0.01. Only about 1% of all possible links in a fully connected network are present. As a trans-regional system combining high-speed and regional connections, the average link number to other stations is 6.64, which is larger than in other city transportation networks (see Table 1). There are five high-degree nodes with more than 100 links, which also obtain the highest values in betweenness centrality. The average path length is 4.23, i.e. a passenger needs to pass a mean of 4.23 stations to reach his/her destination. The longest shortest path, i.e. the diameter, passes along 11 stations. Thus, the railway connections are efficient in such a way that other PTNs exhibit longer path lengths. This observation can be

further supported by relatively large transitivity, i.e. probability for the existence of a third link to close a triangle, which is 0.17.

For our simulation study, we use three different PTNs from the optimization software LinTim (Goerigk et al., 2015, see Section 4): One is similar to the German railway system (see Figure 1), which is a typical representative for general transportation networks (see Table 1). The other two networks, Athens metro and the bus network of the city of Göttingen, exhibit unusual network layouts and hence serve for comparing purposes (see Section 4). The PTN similar to the German railway system contains only stations connecting long-distance traffic. It consists of $K = 319$ nodes connected by 446 links. Thus, it is less detailed than the one used in the application. The average link number to other stations is 2.8 and therefore similar to other networks (see Table 1). There are few stations of high importance with a large number of links in various directions, e.g. the main stations of Mannheim, Hanover and Leipzig (see Figure 1). A more sophisticated node centrality measure is betweenness. Here, Kassel-Wilhelmshöhe, Nürnberg and Fulda gain most importance. The Göttingen bus system is a directed network with 257 stations and 548 connections. It exhibits a strikingly high average degree in comparison to the other regional PTNs reported in the literature (Sienkiewicz and Holyst, 2005; von Ferber et al., 2009, see Table 1). The Athens metro is a rather small network with 51 nodes and 52 links, which results in relatively large density of 0.04, but low average degree of 2.04. Additionally, it is noticeable that this network does not exhibit any triangles which results in a transitivity of zero. In contrast to other PTNs, Athens is a rather extreme case (see Table 1).

In comparison with the food shipping network that was constructed for source detection during food-borne disease outbreaks in Germany (Manitz et al., 2014), we can conclude that PTNs generally seem to be very sparse, with relatively long path lengths and low transitivity (see Table 1).

3 Network-theoretic Methods

3.1 Network-based Source Detection

We enhance the source detection method by Manitz et al. (2014), which was originally suggested for the reconstruction of infectious diseases breaking out from an epicenter (see also Brockmann and Helbing, 2013). We generalize it for the application on universal network-based propagation processes, so that it can be also applied to the spreading of delays in railway systems.

The approach requires an underlying graph $G = (\mathcal{K}, \mathcal{L})$, which can be specified as a collection of nodes $k \in \mathcal{K} = \{1, \ldots, K\}$ that are connected by direct links $(k, l) \in \mathcal{L} = \{(k, l) | k, l \in \mathcal{K}\}$ between them. It consists of only one component, so that any two nodes k and l are connected via a path γ_{kl}, i.e. an ordered sequence of links, in the existing network. When modeling food-borne infectious diseases, the underlying network captures the transportation routes of contaminated

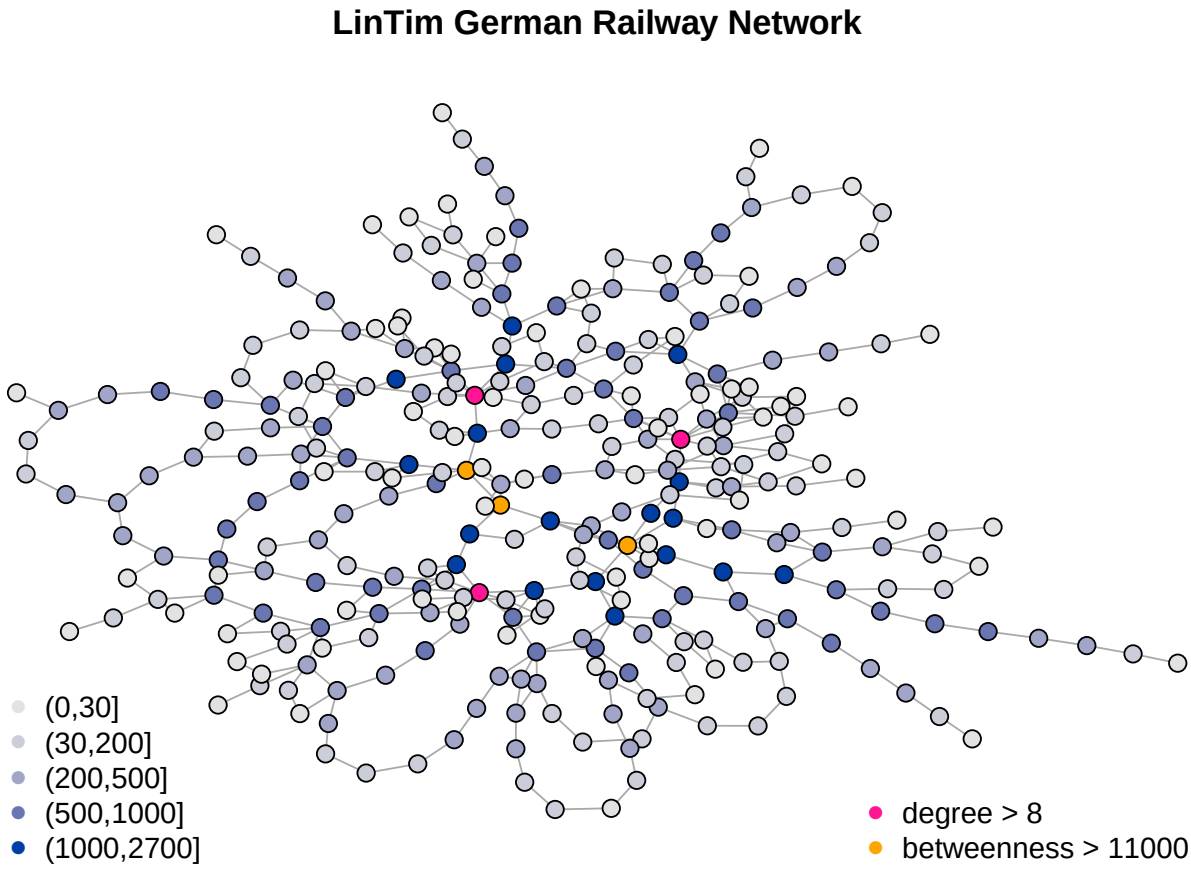


Figure 1: **German high speed railway network** obtained from the optimization software LinTim (Goerigk et al., 2015). Nodes are color-coded according to degree-normalized betweenness centrality and positioned using layout by Kamada and Kawai (1989). Additionally, high-degree and high-betweenness nodes are highlighted in pink and orange, respectively. Betweenness centrality measures the number of shortest paths passing each node. Within the railway network, stations in the margins of the network have low degree-normalized betweenness centrality. These nodes are in particular final stations and connections to the systems of the neighboring countries.

food. In case of train delays the network (PTN) represents the stations and tracks which are used by trains and on which delays are propagated (see Section 2).

An appropriate distance $d(k,l)$ is defined for a path γ_{kl} between nodes $k,l \in \{1,\dots,K\}$ of the network, which will be specified in the next section. Furthermore, we assume a time-dependent stochastic process $\{X_k(t)\}$ on the network nodes $k \in \mathcal{K}$ characterizing a propagation mechanism in a time range $t = 1,\dots,T$. Corresponding observations $x_k(t)$ in each node k are conducted at different time points to find T sequential pictures of the distribution pattern. During the source detection analysis in PTNs, $x_k(t)$ corresponds to the relative delay magnitude observed in a station k within a time slot $[0,t]$, i.e., the total delay in station k, normalized by the number of trains passing this node.

Assuming an appropriate distance definition, which will be defined in the next section, propagation phenomena are spreading in a circular fashion from the origin $k_0 \in \mathcal{K}$ (Manitz et al., 2014; Brockmann and Helbing, 2013). Then, the focal idea of source reconstruction is to test different source candidates and examine the concentricity of the observed pattern on a minimum shortest-path tree with the candidate k_0 as the root. Thus, given an appropriately defined distance d, the source can be reconstructed by minimizing the expected value of the distance $\mu_X(d;k_0,t)$ from the origin k_0 to all other network nodes $k \in \mathcal{K}$ specified by process $X_k(t)$, i.e.

$$\hat{k}_0(t) \in \arg\min_{k_0 \in \mathcal{K}} \mu_X(d;k_0,t). \tag{1}$$

Since $\mu_X(d;k_0,t)$ is continuous, we obtain with probability one a unique solution.

Thereby, the expected distance $\mu_X(d;k_0,t)$ can be estimated by the average distance $d(k,k_0)$ from source k_0 to all destination nodes k weighted by the observed mean magnitude of delays $x_k(t)$ in node k until time t. Thus,

$$\hat{\mu}_X(d;k_0,t) = \frac{1}{N_x(t)} \sum_{k=1}^{K} x_k(t) \cdot d(k,k_0), \tag{2}$$

where $N_x(t) = \sum_k x_k(t)$ is the total relative delay in the network at time t.

3.2 Network-based Effective Distance

For the transformation of the irregular diffusion pattern of modern spreading phenomena into a typical concentric spreading circle it is necessary to replace the classic geographic distance by an appropriate network distance. For the spread of modern epidemics Brockmann and Helbing (2013) introduced the so called, which is required for the success of source detection.

We consider a link weight w_{kl} from node l to k for all nodes $k,l = 1,\dots,K$ of the underlying network. Usually, this corresponds to a physical quantity measuring the linkage strength, e.g. traffic load or interaction intensity. The corresponding transition probability p_{kl} from node l to

k can be derived as the probability for a transit to k when being in node l, i.e.

$$p_{kl} = w_{kl}/n_l, \quad \text{for all } k, l = 1, \ldots, K, \tag{3}$$

where $n_l = \sum_k w_{kl}$ is the aggregated weight of all outgoing links. A transition probability equal to zero means no direct linkage between the nodes.

The effective distance is defined as the effective length of a path, which is a combination of the topological length and the logarithmic path probability, minimized for all possible paths $\gamma_{kl} \in \Gamma_{kl}$ from origin l to destination k (Brockmann and Helbing, 2013). Thereby, the topological length $L(\gamma_{kl})$ is given by the number of links composing the path γ_{kl} along the nodes $l = k_0, k_1, \ldots, k = k_{L(\gamma_{kl})}$. The path probability is the product of the transition probabilities $p_{k_i,k_{i-1}}$ for $i = 1, \ldots, L(\gamma_{kl})$ of the corresponding links in the path $\mathcal{L}_{\gamma_{kl}}$. A path is considered to be short, if the probability of transiting the path is high, i.e.

$$\begin{aligned} d_{\mathtt{eff}}(k,l) &= \min_{\gamma_{kl}\in\Gamma_{kl}} \left[L(\gamma_{kl}) - \log\left(\prod_{(k_i,k_{i-1})\in\mathcal{L}_{\gamma_{kl}}} p_{k_i k_{i-1}} \right) \right], \quad \text{for } k, l \in \mathcal{K}. \\ &= \min_{\gamma_{kl}\in\Gamma_{kl}} \left[L(\gamma_{kl}) - \sum_{(k_i,k_{i-1})\in\mathcal{L}_{\gamma_{kl}}} \log p_{k_i k_{i-1}} \right]. \end{aligned} \tag{4}$$

If the probability of a path equals one, the effective distance is the number of path links, i.e. the deterministic topological distance. The less probable a path is, the larger the effective distance. If the probability for a path approaches zero, the effective distance converges to infinity. Note that, apart from the network, the effective distance depends only on the static transition probabilities p_{kl}, which define the network structure.

On an arbitrary network, the effective distance can be computed by using Dijkstra's algorithm (Dijkstra, 1959). It is an efficient search algorithm, which was developed to solve the single-source shortest path problem for weighted networks. The modification of the inverse link weights as the distance $1 - \log(p_{kl})$ for link $(l, k) \in \mathcal{L}$ results in the effective distance, whereas a transition probability of $p_{kl} = 0$ on a path would result in an infinite distance.

The following methods are for comparison. The first method comes from the public transportation idea, which we developed specifically for propagated delays in PTNs, whereas the second method is rather an idea based on network theoretic analysis of node centrality.

3.3 Recursive Backtracking Algorithm for Delay Propagation

The basic idea of backtracking would be the tracing of delays back in time. Approaches similar to this one can be found in Yamamura et al. (2013) and references therein. Their approach

uses more comprehensive information and is thus only in the principle concept related to this approach. Even if the data on the precise planned and actual times of all trains were available, source estimation would not be an easy task since delays can be propagated via various mechanisms (the same train, passengers or due to track occupation). Even more, an observed delay can be composed of different source and propagated delays. Assuming imprecise recording data increases the difficulty of tracing back delays to their source even more. Additionally, in this setting the given data does not contain information about the actual train rides. Hence, we introduce a backtracking method adapted to the minimal data basis, that makes explicitly use of the way delays are spreading in a PTN:

Given a pattern at time t, let $b_k = 0$ for all $k \in \mathcal{K}$ be a variable counting events in the following:

1. Consider a node $k \in \mathcal{K}$ which has experienced a relative delay $x_k(t)$.
2. Going through all adjacent nodes $k' \in \mathcal{K}$ such that there is $(k, k') \in \mathcal{L}$, we look for the next node with the highest relative delay

$$k^* \in \arg \max_{k' \in \mathcal{K}:(k,k') \in \mathcal{L}} x_{k'}(t). \tag{5}$$

3. If the relative delay magnitude of this new node is higher than the one of the first node ($x_{k^*}(t) \geq x_k(t)$), we jump to the new node and repeat ($k := k^*$; go to 2.).
4. This process is executed until the current node relative delay magnitude is higher than all of its adjacent nodes $\big(x_k(t) > \max_{k' \in \mathcal{K}:(k,k') \in \mathcal{L}} x_{k'}(t)\big)$. In this case we increase b_k by one.

The loop (1.-4.) is repeated for all nodes $k \in \mathcal{K}$ which have experienced delay $x_k(t) > 0$. Finally, all nodes $k \in \mathcal{K}$ are ranked by b_k which gives the number of times that such an iteration ends at each node.

3.4 Source Detection Approach based on Node Centrality

Comin and da Fontoura Costa (2011) suggested that the starting point of a spreading process can be reconstructed as the network node that obtains the highest centrality in the transmission tree. The node centrality is measured by node betweenness normalized by the corresponding node degree. We adapt this approach to the presented data availability, where no transmission tree is given. In the application to the delay case, we consider the subgraph induced by all nodes which are affected by delays. From this subgraph the node with the highest betweenness value is detected to be the source node. For this, we require that at least four stations have experienced delays to ensure a minimal spreading pattern.

4 Delay Simulation

4.1 Research Hypotheses

Here we discuss the research hypotheses, which will be analyzed in the delay simulation study.

1. **Applicability of Method:** Since the mechanism of spreading of infectious diseases is similar to the propagation of delays, source detection methods are assumed to be applicable. Good performance quality would indicate similar underlying propagation mechanisms.

2. **Time Increase:** In general, the methods apply to spreading patterns that break out in an epicenter and spread through both time and space. As the pattern's diversity increases in time, the method's reliability is expected to decrease. This is a natural hypothesis as the spreading is assumed to be a stochastic process which includes probabilistic propagation.

3. **Node Centrality:** The reliability of the methods are expected to increase for larger network centrality of the epicenter. Source nodes having high node centrality cause a high number of propagated delays spreading along various directions through the network. Hence, detection methods are expected to be more reliable for more central nodes.

4. **Robustness:** The quality of the detection method can be exhibited by its robustness. It is expected that neither consideration of track occupation in the timetable nor different delay management strategies have significant influence on the performance quality. Since different delay management strategies result in different underlying propagation mechanisms, similar performance would approve the robustness of our source detection approach.

5. **Network Structure:** Considering the network layout, we expect the method to work better given a well-connected network. The less connected a network layout is in comparison to other world transportation networks, e.g. in terms of transitivity, we expect the delay pattern to be less regular and with that the source to be less well predictable.

6. **Realistic Data:** The method can only be helpful if it proves to be effective in realistic settings. Indeed, we expect the method to work reasonably well on realistic instances which include noise. If the noise reaches a level at which the signal is not distinguishable anymore the method is not expected to return reliable results. The consideration of noise is supposed to mimic real data, but in an experimental set-up.

4.2 Delay Simulation Data

We test the network-based source detection approach in comparison to the recursive backtracking algorithm and the adapted centrality-based method by Comin and da Fontoura Costa (2011) in various simulation settings. Basis for this are diverse delay spreading patterns simulated with

LinTim (Goerigk et al., 2015), which is a scientific software toolbox giving the possibility to solve the various planning steps in public transportation.

4.2.1 Line Planning and Timetabling

Using LinTim, we generate a line plan based on a specific PTN and estimated traffic loads for the network links. The corresponding optimization problem solution includes the line paths and their frequency (see Schöbel, 2012, for an elaborate overview on the line planning problem).

In the next step, a timetable of four hours is generated (based on an Event-Activity-Network, see Serafini and Ukovich, 1989; Nachtigall, 1998; Schöbel, 2007). A timetable is given by the arrival and departure times of all trains at all stations. It is hence the published plan on which a company operates its schedule.

4.2.2 Generation of Source Delays

Unpreventable occurrences lead to delays in a given timetable which raise the need to deal with them. We distinguish between so-called *source* (or *initial*) and *propagated* delays.

In each run of a simulation scenario, one of the $K = 319$ stations of the network represents the epicenter of the delay. For each simulation, we generate 30 source delays of a randomly determined magnitude in one particular station of the PTN. Allowing only 30 source delays – even though more trains might pass the source node – ensures comparability among the different nodes. The delay magnitude is drawn from a negative exponential distribution with a mean value of 300 seconds. This distribution function has been reported to give realistic outcomes (Vansteenwegen and Van Oudheusden, 2006; Goverde, 1998; Meng, 1991).

For the simulation of noise delay we additionally consider delay drawn from the same distribution with mean value of 60 seconds. In order to see the relation between the detection performance and the amount of noise, the percentage of affected activities was varied between 1% and 50%.

4.2.3 Propagation of Delays

The other type of delays are so called *propagated* delays. Obviously, the malfunction of the ride of some trains can affect other trains as well. We consider train dependencies, and thus decisions that have to be taken, of two different kinds.

- *Wait-Depart Decisions:* There are passengers who want to transfer from one train to another to reach their destination. In case the first train arrives late at a station where passengers want to transfer to another train, the operator has to decide if the connecting train should wait for transferring passengers or depart on time. In the first case, the delay is propagated from the first to the second train.

- *Sequence Decisions:* Delays are also spread because of limited track capacity and security distances between trains. A given timetable specifies the sequence in which trains pass a track. If a train arrives late at the beginning of a track, it has to be decided, if other trains should overtake or not. Maintaining the sequence as planned results in a knock-on delay for the other trains. Switching the sequence might increase the delay of the first train even more.

Other dependencies such as vehicle schedules or crew schedules are neglected in this study. The *delay management problem* consists then of the decisions about maintaining passenger transfers and train sequences such that the passenger delay is minimized (for an extensive discussion see Schöbel, 2001; Schachtebeck and Schöbel, 2010; Schachtebeck, 2010). Practitioners usually use strategies based on fixed waiting time rules, e.g. a train waits for transferring passengers from a delayed train only if the delay is below a fixed time. Those rules are easy to remember but there are more sophisticated heuristics. We apply rules that can be grouped into the following two groups (for details see supplementary materials):

- *Group 1 (one-step optimized)* In this group the rules only decide upon transfers (i.e. wait-depart decisions). The track occupation (i.e. sequence decisions, also called *headways*) are fixed according to a given timetable. To this group belong three different fixed waiting time rules, and two heuristics solving an integer program. These rules are called rules 1–5, respectively.

- *Group 2 (two-step optimized)* The rules in this group determine both the transfers from the wait-depart decisions as well as the track sequencing from the sequence decisions. The three heuristics belonging to this group determine the sequencing in a first step and find the explicit times for all arrivals and departures in a second step of optimization. The rules only differ in the way they deal with the transfers. These rules will be referred to as rules 6–8.

4.2.4 Data Recording

Using a chosen delay management strategy and the generated source delays, we propagate the delay through the network. We conduct observations of the propagated delays at ten equally distributed time points to find sequential pictures of the propagation pattern. Each of them gives a snapshot of a delay dispersal, while the sequence of the pictures depicts the spreading process in space and time.

For each time slot, the data for delay source reconstruction is gathered as follows: Every time a train arrives or departs late at a station, the delay magnitude for that station is measured. The total delay magnitude is normalized by the number of trains passing this station. Note that the delay both at the arrival and at the departure are considered for the corresponding station.

4.3 Performance Evaluation

The performance is quantified using four different performance measures: probability of correct detection, rank of correct detection, shortest distance to correct detection and standard deviation of distance to correct detection.

Probability of correct detection The main concern of source detection performance is the correct estimation of the source. Thus, we quantify source detection performance by probability of correct detection, which is given by the relative number of correct source detections. A higher probability of detection indicates a better performance.

Rank of correct detection Secondly, we examine the rank of the correct source node. To obtain the rank of correct origin, we order all source candidates k_0 according to their average effective distance $\hat{\mu}_X(d; k_0, t)$ as given in Equation (2; see Section 3). Obviously, a low ranking means good performance, while a rank of one is equal to correct detection.

Distance to correct detection Thirdly, we study the distance to correct detection, which we define as the distance to the true source node on the network. More precisely, we find the shortest path from the estimated source $\hat{k}_0(t)$ to the correct source node $k_0(t)$ and aggregate the track lengths (in km or traveling time) of the links on the path. The lower the distance to correct detection, the better the performance of source detection.

Standard deviation of distance to correct detection Finally, we analyze the standard deviation of the distance to correct detection to examine the variability in the results of the approach. This gives information about the underlying distribution of the source detection performance.

Statistical model analysis For deeper analysis, we fit a statistical model, which simultaneously analyzes the probability of detection π_i and the expected distance to the node of correct detection μ_i in dependence of the covariates time t_i and source node centrality $c_i(k_0), i = 1, \ldots, n$.

Since distance is a continuous positive variable with left-skewed distribution, we consider the corresponding observations $y_i, i = 1, 2, \ldots, n$, to be Gamma distributed, i.e.

$$y_i | \mu_i, \nu_i \sim \text{Gamma}(\mu_i, \sigma), \quad i = 1, 2, \ldots, n$$

where $\mu_i > 0$ is the expected distance and $\sigma > 0$ describes the standard deviation. Usually, the Gamma distribution has positive support and is used to model waiting times and other left-skewed responses.

We want to treat observed distances with length zero, i.e. correct detection, as a special case. Thus, the zero-adjusted Gamma distribution should be used. Then, the probability

density function $f_{dens}(y_i|\pi_i, \mu_i, \sigma)$ is conditional on the parameters π_i, μ_i, and σ corresponding to probability of detection, expected detection distance, and standard deviation, respectively.

$$f_{dens}(y_i|\pi_i, \mu_i, \sigma) = \begin{cases} \pi_i & , \text{if } y_i = 0 \\ (1-\pi_i) f_{\text{Gamma}}(y_i|\mu_i, \sigma) & , \text{otherwise.} \end{cases}$$

The specific data structure can be captured by a Generalized Additive Model for Location, Shape and Scale (GAMLSS; Rigby and Stasinopoulos, 2005). Beside the mean expectation it is also able to model effects for location, shape and scale. Since GAMLSS belongs to the semi-parametric regression-type models, the covariates can be modeled non-parametrically to capture flexible non-linear effects, i.e.

$$\begin{aligned} \text{logit}(\pi_i) &= \beta_\pi + s_{\pi 1}(t_i) + s_{\pi 2}(c_i(k_0)), \\ \log(\mu_i) &= \beta_\mu + s_{\mu 1}(t_i) + s_{\mu 2}(c_i(k_0)), \end{aligned}$$

where the parameter β_π and β_μ are the intercepts for each of the submodels. Furthermore, $s_{\pi 1}, s_{\pi 2}, s_{\mu 1}$ and $s_{\mu 2}$ are non-linear effects approximated by P-splines (Eilers and Marx, 1996). The covariates are specified by the predefined research hypotheses, so that model selection is redundant. The model is fitted using the R package `gamlss` (Stasinopoulos and Rigby, 2007; R Core Team, 2014).

4.4 Results

4.4.1 Detection Performance

In the standard scenario, we simulate spreading patterns, where security distances between trains are ensured and the delay management rule 2 (one-step optimized, group 1) is applied. Source detection performance of the network-based method with relative delay magnitude and delayed train counts in dependence of time is compared to the performance of backtracking and the Comin's centrality-based approach. Figure 2 depicts the results of this analysis.

The results reveal, that backtracking achieves principally the highest proportions of correct detections (84.3% at time point 2 decreasing to 53.3% at time point 10). The network-based as well as the backtracking source detection cannot be performed at the first two time points for some of the simulated patterns (22.9%/2.2% instances with no delay information for delay magnitude; 19.1%/1.3% instances with no delay information for delayed train counts), because initial delays were not yet generated. At the third time point, the correct source can be roughly equally often reconstructed when delay magnitude or delay train counts are given (74.9% or 77.4% of the simulations, respectively). The detection rates diverge with increasing time, so that at the last time point 47.6% and 21.6% of the pattern sources, respectively, are correctly detected. The performance of Comin's source detection approach is considerably diminished

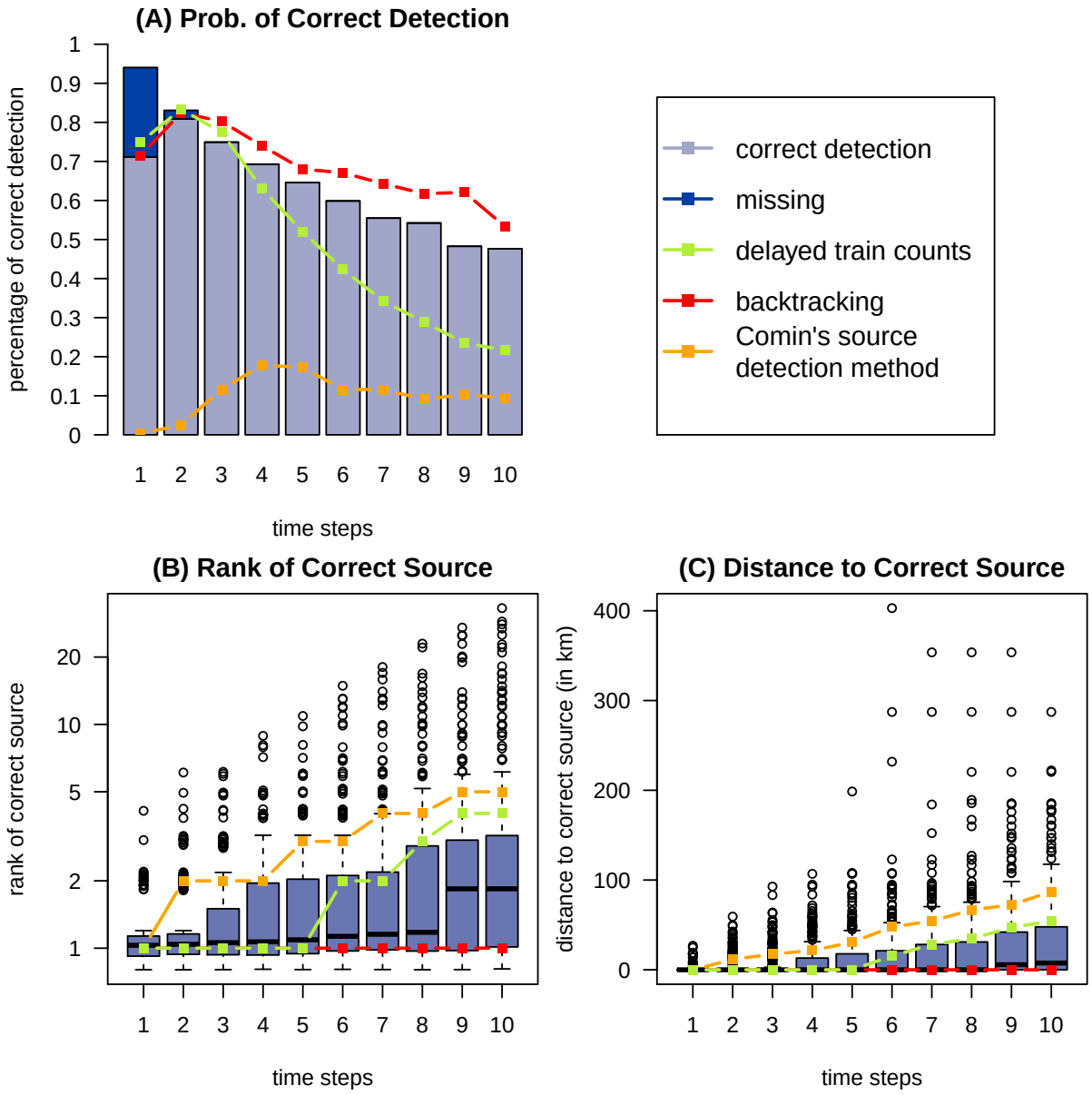


Figure 2: **Detection performance in the standard scenario (with consideration of security distances and delay management rule 2).** (A) Percentage of correct detection, (B) rank of correct detection, and (C) distance to correct detection (in km) over time. The percentage of correct detection is compared to the results from network-based source detection with the number of delayed trains, recursive backtracking, and the source detection approach by Comin and da Fontoura Costa (2011).

by a large amount of missing instances (99.7% at $t = 1$, 62.4% at $t = 3$, decreasing to 3.1% at $t = 10$) due to the minimal delay spreading required. Maximum detection rate of 17.9% is reached at the fourth time point.

Accordingly, the median of the source detection rank for the recursive backtracking approach is one for all time points with maximum rank values of 35. In comparison, the median rank for the network-based method remains at one for the first eight time points and increases after that to two, but the maximum always remains below 34. If only the number of delayed trains is known, the median increases already at $t = 6$ to two and reaches four at the last observation (maximum rank 50). This indicates high performance of both source detection methods considering the maximum possible rank of $K = 319$. Comin's method does not perform of the same quality as only in time point one a median of one is achieved whereas for all other time points medians of at least two are obtained (maximum rank 143 at $t = 10$).

The backtracking method consistently achieves a median of 0 for the distance to correct detection but the mean is varying more. For the network based method, the median distance of correct detection varies from 0 km in the beginning to 7.52 km and 54.39 km at the last observation for relative delay magnitude or delayed trains given, respectively. The last value is comparable to the mean track length (50.23 km) of a link. The median distance to the correct source is 86.97 km when applying the centrality-based source detection approach. Obviously, the occurrence of very distant estimations is more likely at the later time points. For the rank of correct detection as well as for the distance to correct source, we observe increasing variability in the detection performance with ongoing time.

The results suggest the general applicability of the network-based source detection approach (research hypothesis 1), which indicates that train delay spreading has some similar underlying propagation mechanism as the transmission of infectious diseases. Furthermore, the results indicate decreasing performance with increasing time steps (research hypothesis 2). Note that generally better detection performance is ensured if the relative delay magnitude instead of the number of delayed trains is taken into account. Comparing the network-based source detection method to the backtracking approach we observe a slightly worse performance. This can be explained by the fact that backtracking is specifically engineered for delay propagation mechanisms. It can hence be considered as an upper bound of source detection methods for universal applications.

The influence of time and node centrality on the performance of the network-based source detection (research hypotheses 2 and 3) are further analyzed by a statistical model as introduced in Section 4.3. According to the centrality-based source detection, we define node centrality as the ratio of betweenness and node degree. The resulting smooth functions describe the effects of the covariates on the probability for a correct detection and the detection distance if the source detection has failed (see Figure 3).

The curves exhibit that the probability of detection decreases over time. At the first time

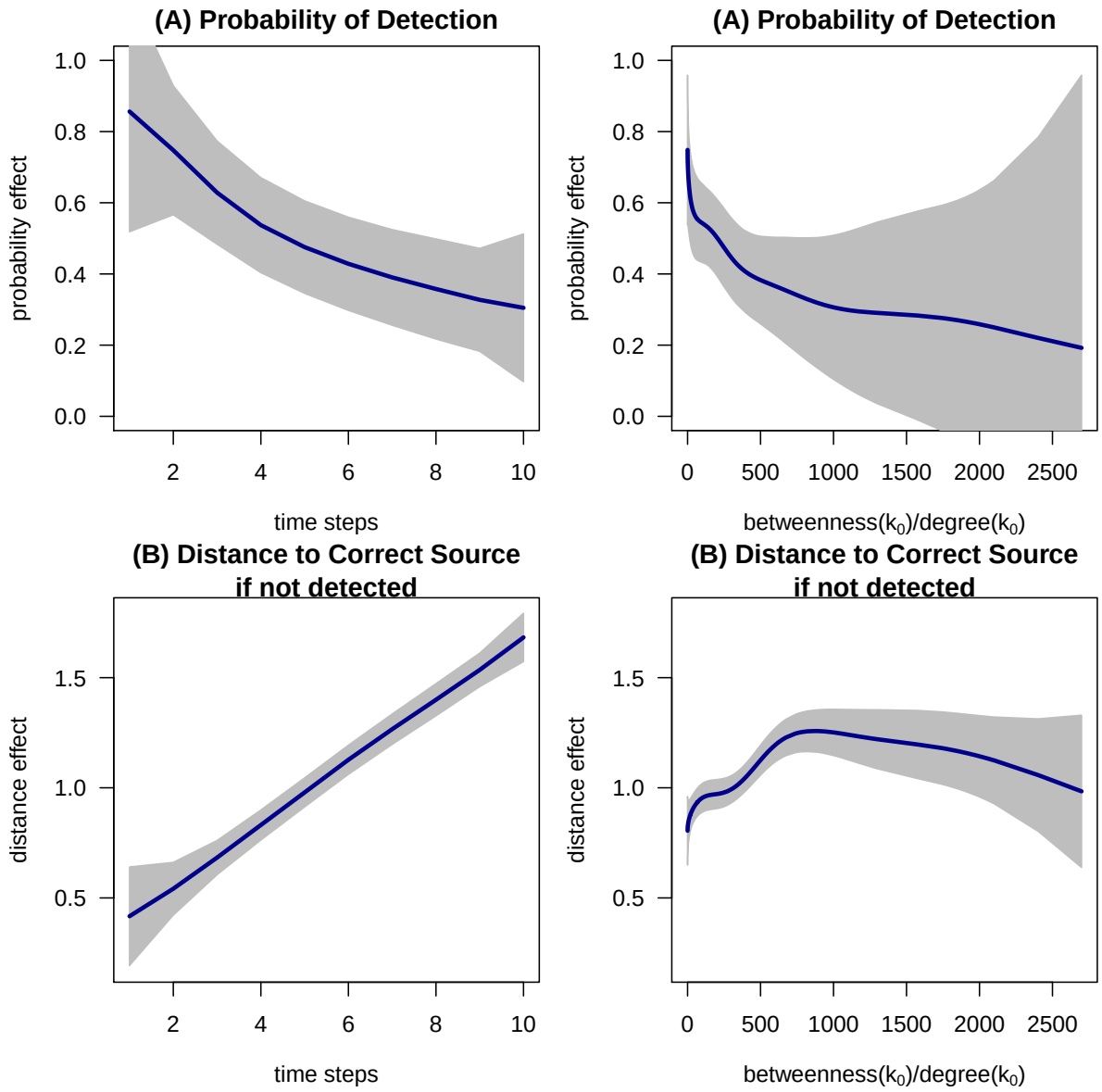


Figure 3: **Smooth Model Effects for zero-adjusted Gamma GAMLSS.** Source detection performance analysis in dependence of time and the degree-normalized betweenness value for source nodes. Effects for (A) probability of correct detection, and (B) distance to correct source (in km) if the source is not correctly detected.

point, 85.0% of the sources are expected to be correctly detected, which decreases over time to 30.4%. There is also a negative influence by the centrality of the source node on the probability of correct detection, so that more central nodes have slightly lower chances of being detected. However, the corresponding confidence intervals are very wide so that this effect cannot be verified.

Whenever the source detection failed, the distance to the true source was modeled assuming a Gamma distribution in dependence of time and source node centrality. The detection distance increases linearly with ongoing propagation of the initially generated delay. However, the distance remains small so that at the last observed time point, we expect a distance of about 1.68 km. The source node centrality shows a relatively weak effect on the detection distance, so that the source estimates are slightly biased if the pattern originates from nodes with moderate centrality.

Altogether, for both parts of the model, we can see an impact of time to the source detection performance, so that research hypothesis 2 can be approved. A considerable influence of node centrality cannot be approved, because effects in the model are relatively weak (research hypothesis 3).

4.4.2 Source Detection for Different Propagation Processes

In this section, we investigate the performance of source detection for various propagation processes, which result in different patterns of delay spreadings. It gives insights into the robustness of the source detection approach to different propagation mechanisms (research hypothesis 4). For that, we simulate the propagation of delays with different delay management strategies (see Section 4.2.3, for details see supplementary materials).

For network-based source detection, the performance can be found to give similar results for various scenarios of delay patterns (see Figure 4), while for recursive backtracking larger differences for the different delay management strategies are obtained. The results show similar trends, while the quantity of correct detections exhibits differences. With ongoing dispersal the gaps enlarge. E.g. in the beginning of source detection for the network-based method the proportion of correct detection ranges between 92.3% and 93.1%, which decreases until the observation at time $t = 10$, when the proportions are between 35.7% and 58.0%. In comparison, the proportions of correct detection for backtracking are between 92.3%% and 93.5%% at the first time point and between 49.8%% and 70.2%% at $t = 10$. Both methods rank the correct source on average within the top 5 (network-based source detection below 4.4, backtracking below 3.2). The shortest distance to the correct source is less than 43 km for the network-based approach and below 66 km for recursive backtracking. Also the standard deviation of the distance to correct detection is very similar for all delay management strategies. For network-based source detection, it increases over time and ranges between 34.4 and 53.6. For backtracking

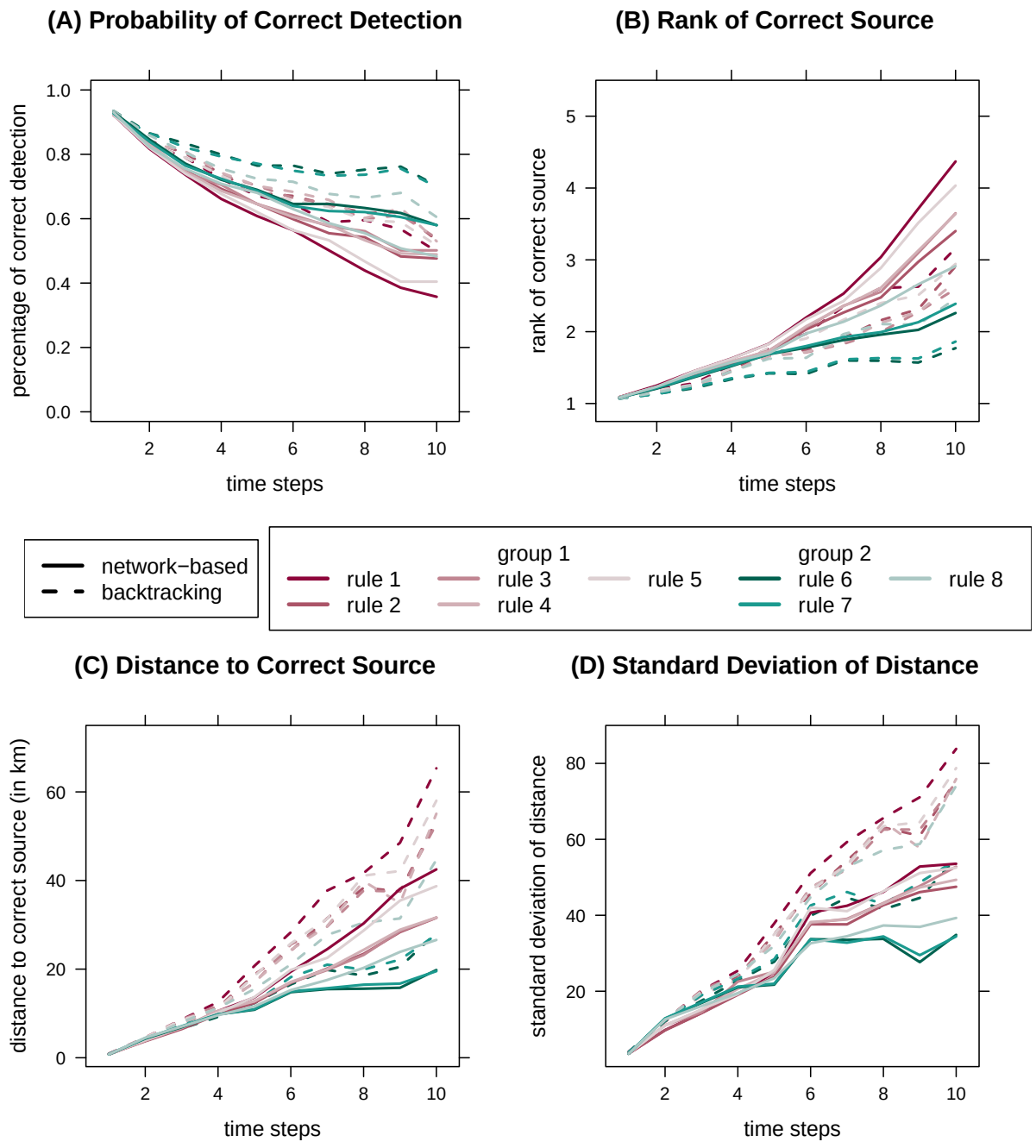


Figure 4: **Comparison of detection performance in simulations with consideration of track occupation for different delay management strategies.** (A) Percentage of correct detection, (B) rank of correct source, (C) distance to correct source (in km), and (D) standard deviation of distance to correct detection over time.

the standard deviation is generally higher and ranges between 54.6 and 78.8 at the last time point.

Considering the quality of detection, we can distinguish the two groups of delay management strategies (see Section 4.2.3, for details see supplementary materials). Source detection for patterns with a delay management strategy solving the delay management problem in a two-stage fashion (group 2) seems to perform better than for methods that optimize only once (group 1). For instance, strategy 8 is a two-stage extension of rule 5 but both strategies take passenger transfers into account. For both source detection approaches, we can observe consistently better performance on patterns based on rule 8 than on rule 5 (e.g. for $t = 10$: network-based method 48.3 vs. 40.4%; backtracking 60.5% vs. 51.7%). However, the source detection works best for the patterns produced with rule 6 or rule 7. The corresponding percentages of correct detection remain over 58.0% (for backtracking even 69.9%), the average correct source rank remains below 2.4 (for backtracking 1.9), while the estimated source is very close to the correct source (average distance below 20 km; for backtracking below 76 km).

Table 2: **Total relative magnitude of delays in the system and proportion of correct detection** obtained by network-based source detection and recursive backtracking at the last time step $t = 10$ differentiated for different delay management strategies (for details see supplementary materials).

delay management		mean delay magnitude	percentage of correct detection	
			network-based	backtracking
group 1	rule 1	169.95	35.7%	49.8%
	rule 2	124.49	47.6%	49.8%
	rule 3	115.67	50.2%	53.9%
	rule 4	122.61	48.9%	53.0%
	rule 5	152.90	40.4%	51.7%
group 2	rule 6	86.52	58.0%	70.2%
	rule 7	89.48	58.0%	70.0%
	rule 8	113.01	48.3%	60.5%

Further inspection of the results revealed a strong association between the total magnitude of delays in the network and the proportion of correct detection. The different delay management strategies are variable in their success of delay containment. Thus, we computed total magnitude of delays in the system at time step $t = 10$ averaged for all source simulations distinguished by delay management strategy (see Table 2). The correlation between percentage of correct detection and mean magnitude of delays (Spearman's correlation coefficient $\rho = -0.991$) exhibit an almost perfect negative association for the network-based source detection approach ($\rho = -0.869$ for recursive backtracking). Thus, the performance differences between the scenarios for

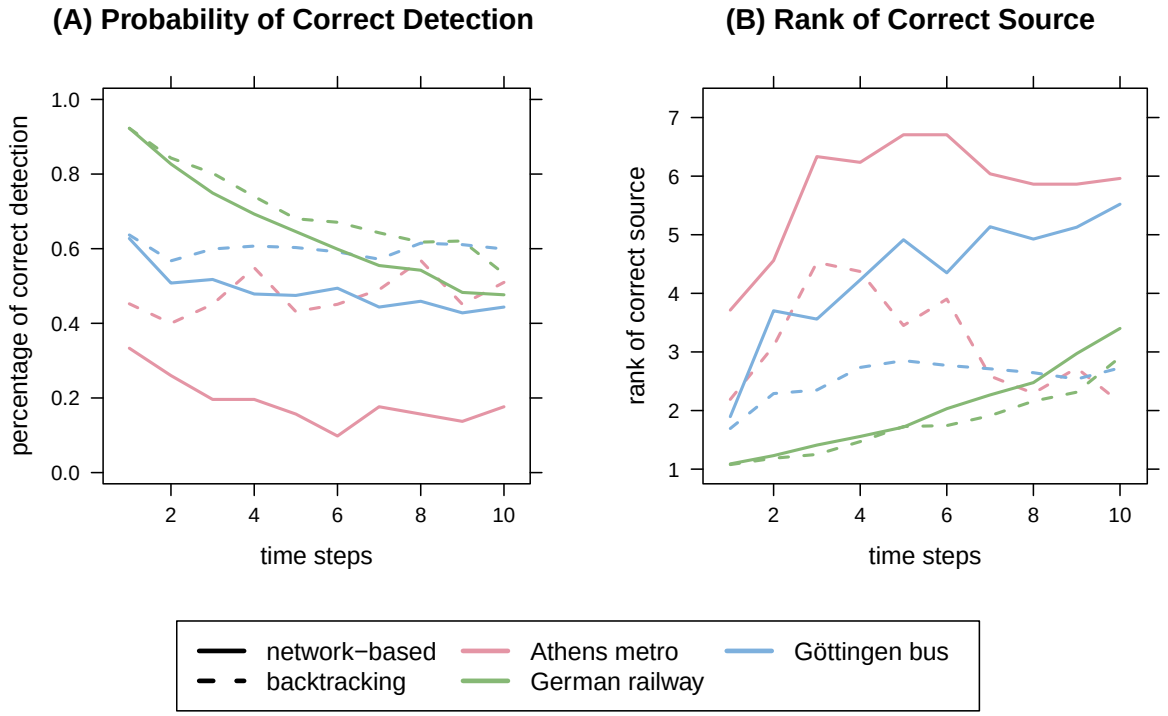


Figure 5: **Comparison of detection performance in simulations on different public transportation systems.** (A) Percentage of correct detection, and (B) rank of correct source over time.

the delay management strategies can be explained to a large extent by the amount of delays in the system.

Altogether, it can be shown that our approaches for source detection are extremely robust regarding different propagation mechanisms (research hypothesis 4). However, we could observe a strong dependency on the total relative delay magnitude in the system, so that more delays result in lower performance.

4.4.3 Comparison for Different Public Transportation Networks

In this analysis, we compare the outcome of the method for different network layouts. In particular, we compare the source detection performance on the German railway network to the results from the networks of the Athens metro and the Göttingen bus. As described in Section 2, the networks have different layouts. In more detail, the Athens metro contains no triangles. In contrast, the networks of the buses in Göttingen and the German railway have reasonably many cycles compared to other world transport networks.

For both methods, best source detection performances can be found for the German railway network, followed by the Göttingen bus network and the Athens metro system (see Figure 5).

At time step three, where for all test instances delays have been recorded, the network-based method reconstructs 74.9% of sources in the German railway correctly. For Göttingen bus only 51.8% and for Athens metro 19.6% of the sources can be correctly determined. In contrast, with the recursive backtracking approach 80.3% of sources can be correctly identified in the German railway network, where 60.0% on the Göttingen bus network and 45.1% are found in the Athens data set. Similar tendencies are exhibited by the results in terms of the rank of correct source, where the mean ranks range between 1.0 and 2.4 for the German railway system (between 1.1 and 2.9 for backtracking), between 1.9 and 5.5 for the Göttingen bus network (between 1.7 and 2.9 for backtracking), and between 3.8 and 6.7 for the Athens metro (between 2.2 and 4.5 for backtracking).

This evaluation indicates that for more complex networks, both methods work more reliable (research hypothesis 4). The simpler the structure (see Athens metro) the less effective proves the network-based source detection method. Still, in the simpler cases the recursive backtracking approach clearly outperforms network-based source detection. However, it can be argued that this is no flaw for the network-based method as such a source detection tool gains importance with more complex PTNs, because the pattern of delay spreading are more difficult to observe. Hence, this method gives a tool for analyzing such hidden patterns (research hypothesis 5).

4.4.4 Considering Background Noise

In realistic spreading patterns, we always observe different levels of background noise, meaning small delays triggered at stations which are not considered to be the major source of delay. In this section, we analyze the suggested source detection approaches with respect to their sensibility to differentiate between noise and signal delays. Additionally to the delay pattern in the standard scenario, we superimpose delay on 1% to 30% of all remaining activities. For instance, delaying 1% of the remaining activities corresponds to almost 200 activities (in contrast to 30 source delay activities), where random noise is drawn from an exponential distribution with mean value of 60 seconds.

For both methods, we can see a clear correspondence between the amount of activities that are delayed and the detection performance (see Figure 6). When delaying 1% of the remaining activities, the percentage of correct detection at time point three only drops to 66.8% (without noise: 74.9) when applying the network-based source detection. As the noise increases to 5% (10%, 20%, 30%) the percentage of correct detection decreases to 29.8% (8.5%, 1.3%, 0.9%). At the last time point we obtain a percentage of correct detection of 39.2% for 1% of activities delayed (without noise: 47.6%). With larger amounts of superimposed noise the proportion of detection quickly decreases. In comparison, the recursive backtracking approach yields a little higher proportion of detection, and remains more stable at later time points. At time points later than $t = 4$, we obtain larger detection rates, when noise is considered. For 1% noise

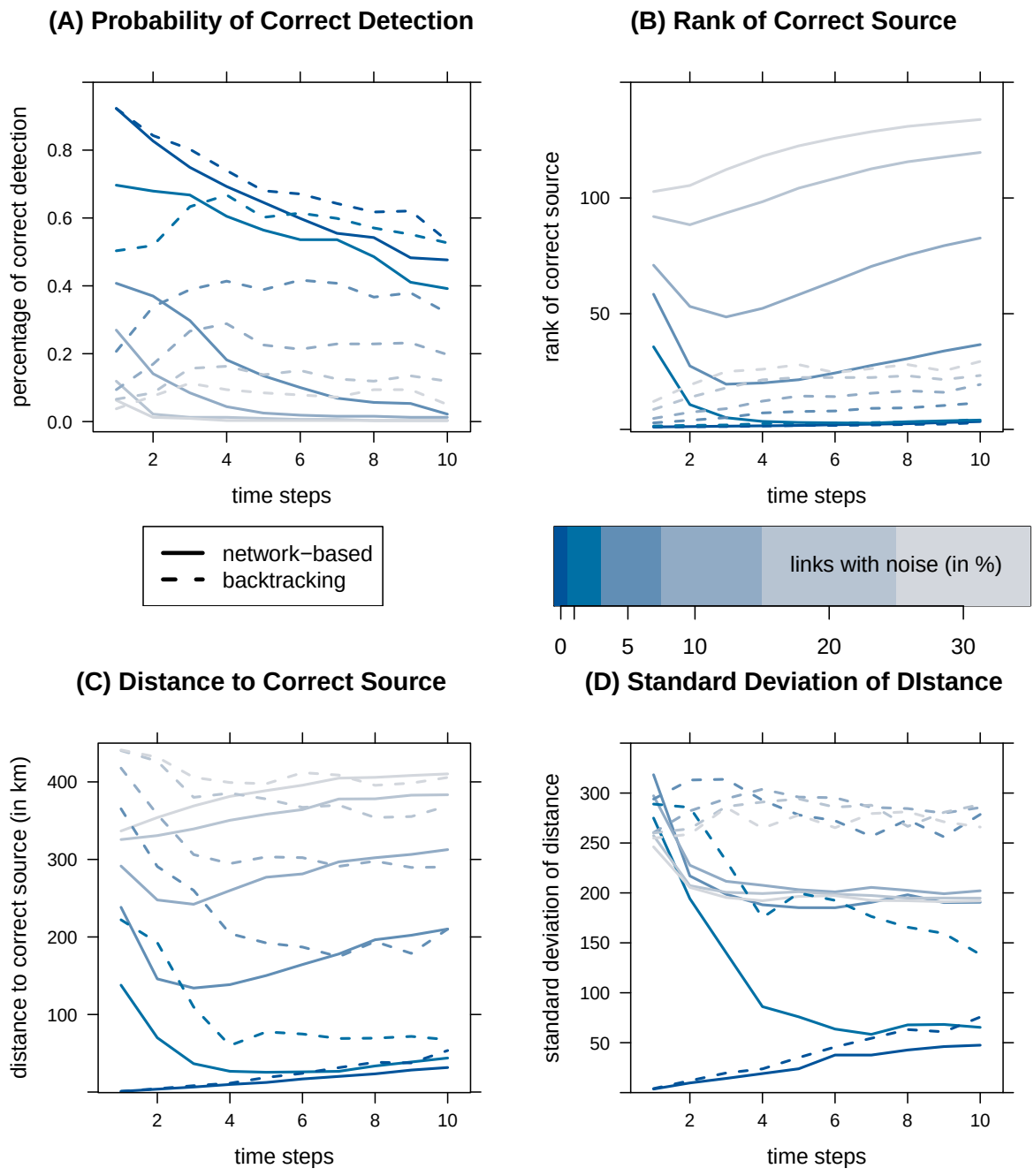


Figure 6: **Comparison of detection performance in simulations of signal-noise scenarios.** (A) Percentage of correct detection, (B) rank of correct source, (C) distance to correct source (in km), and (D) standard deviation of distance to correct detection over time.

66.8% of the sources are detected at $t = 4$, whereas for 5% noise 41.4% at $t = 4$. The same message can also be obtained when considering the other performance measures. The rank of correct source is in general much lower when applying the backtracking approach (average values remaining generally below 30), whereas the distance to the correct origin can be found to be slightly lower for network-based source detection. For the network-based source detection, the standard deviation decreases over time and levels off after the second time step (standard deviation between 191 and 318 for more than 1% noise). We assume that there are only a few signal delays at the beginning, so that the delay signal remains in the background. In comparison, the backtracking approach is less stable and results in higher standard deviations between 255 and 313 for more than 1% noise.

This study shows the general applicability of the method to real world examples and its bounds (research hypothesis 6). Indeed, although noise is included in the data, the detection performance maintains a high precision. Only when superimposed noise is taking high values, the actual signal is not recognizable anymore. Additionally, under consideration of noise, we recognize that detection performance first decreases and then increases as the signal comes to the fore.

5 Application: Train Delays on German Railway System

In collaboration with the department for *Transportation Network Development and Transport Models* of the German railway operator *Deutsche Bahn*, three examples of delay spreading events on the German railway system were selected to apply the developed source detection methods to real data. We compare the results from the network-based source detection and the recursive backtracking with the ones from the centrality-based source detection (Comin and da Fontoura Costa, 2011) using the evaluation measures defined in Section 4.3.

The provided railway network consists of 1049 stations connected by 3484 links, which contains also regional connections (for more details see Table 1). However, for this analysis only the train delays for high-speed trains are available, so that we only have information on delays for a subnetwork with less than 300 nodes (depending on the example). After 6am, the data is aggregated for 30 minutes time slots day-long. The total delay magnitude of all delays is normalized by the number of trains passing the station. Due to confidentiality, we use anonymized station identifier, but provide the degree and betweenness centrality c_D and c_B, respectively.

5.1 Source Detection Results

In the first example, the delays actually originate in station *x.152* ($c_D = 144, c_B = 131,897$), where a damage was recorded at 6:25am. We have delay data for 289 stations. The network-based source detection approach performs well and locates the actual source at 9.30am, about three hours after the damage report (see Figure 7 and supplementary materials). Before that,

another station with id *x.105* ($c_D = 113, c_B = 115,020$) is estimated to be the source of the delay pattern in 126 minutes travel time distance. This station is known to introduce large disturbance into the system. In comparison, the recursive backtracking method detects the true origin or a station ($c_D \in [15; 144], c_B \in [1,068; 131,897]$) in the near proximity instantly for four hours after the damage incidence. However, the estimation is less stable in its reliability. After 10:30am, its prediction yields to an area which is known to be a large tunnel construction site. The centrality-based source detection approach by Comin and da Fontoura Costa (2011) ranks the true origin of delays within the top-10 throughout the day, but does not identify the source. However, the method finds stations which are smaller and more than 200km far from the correct detection ($c_D \in [5; 18], c_B \in [4,245; 15,237]$).

In another example, delays in 297 stations are reported to be caused in station *x.95004823* ($c_D = 36, c_B = 14,671$) from 10.30am due to an electricity failure in a switch tower. The network-based source detection method is able to identify the station which is closest to the damaged switch tower at 2pm, three hours after the damage (see supplementary materials). Before that, another large high-influence station (*x.105*, $c_D = 113, c_B = 115,020$) is considered to be the source, while the true origin is ranked within the top-10. In contrast, the recursive backtracking approach identifies the station of origin (or near proximity; $c_D \in [16; 55], c_B \in [464; 30,189]$) very quickly, already 30 minutes after the occurrence. After four hours the spreading pattern becomes blurred and a station 81 km far is estimated ($c_D = 20, c_B = 1,160$). The actual station is usually ranked within the top-30. The centrality-based source detection approach does not detect the correct origin. It is ranked between 38 and 45, while the estimates are far away (between 300 km and 1100 km; $c_D \in [4; 6], c_B \in [1522; 7267]$).

Finally, we analyzed a delay spreading pattern (recorded data for 299 stations) originating from station *x.105* ($c_D = 113, c_B = 115,020$) from 7:45am. The network-based source approach is able to correctly identify the source at 9:30am, where the estimate remains for the rest of the day (see supplementary materials). Backtracking ranks the correct origin between 16 and 31, but identifies a station close-by ($c_D = 58, c_B = 20,786$), which is passed by the majority of the long-distance trains after passing the delay-causing failure. The centrality-based approach by Comin and da Fontoura Costa (2011) neither ranks the correct source very high nor detects a station in the close proximity ($c_D \in [4; 5], c_B \in [1522; 7267]$).

Altogether, the application of the suggested methods to the three real-world examples of delay spreading give reasonable results and show the general applicability of the methods on real data. The network-based source detection seems to be a robust approach, which is reliable in detecting the origin about three hours after the introduction of disturbance into the system. In contrast, backtracking is a highly sensible approach. It is fast in targeting the area of source, but has difficulties to settle on a particular station, which makes the method somewhat instable. After about four hours the correct detection in the area vanishes. The centrality-based source detection is a less reliable source detection approach. In all examples, we recognize the station *x.105* likely

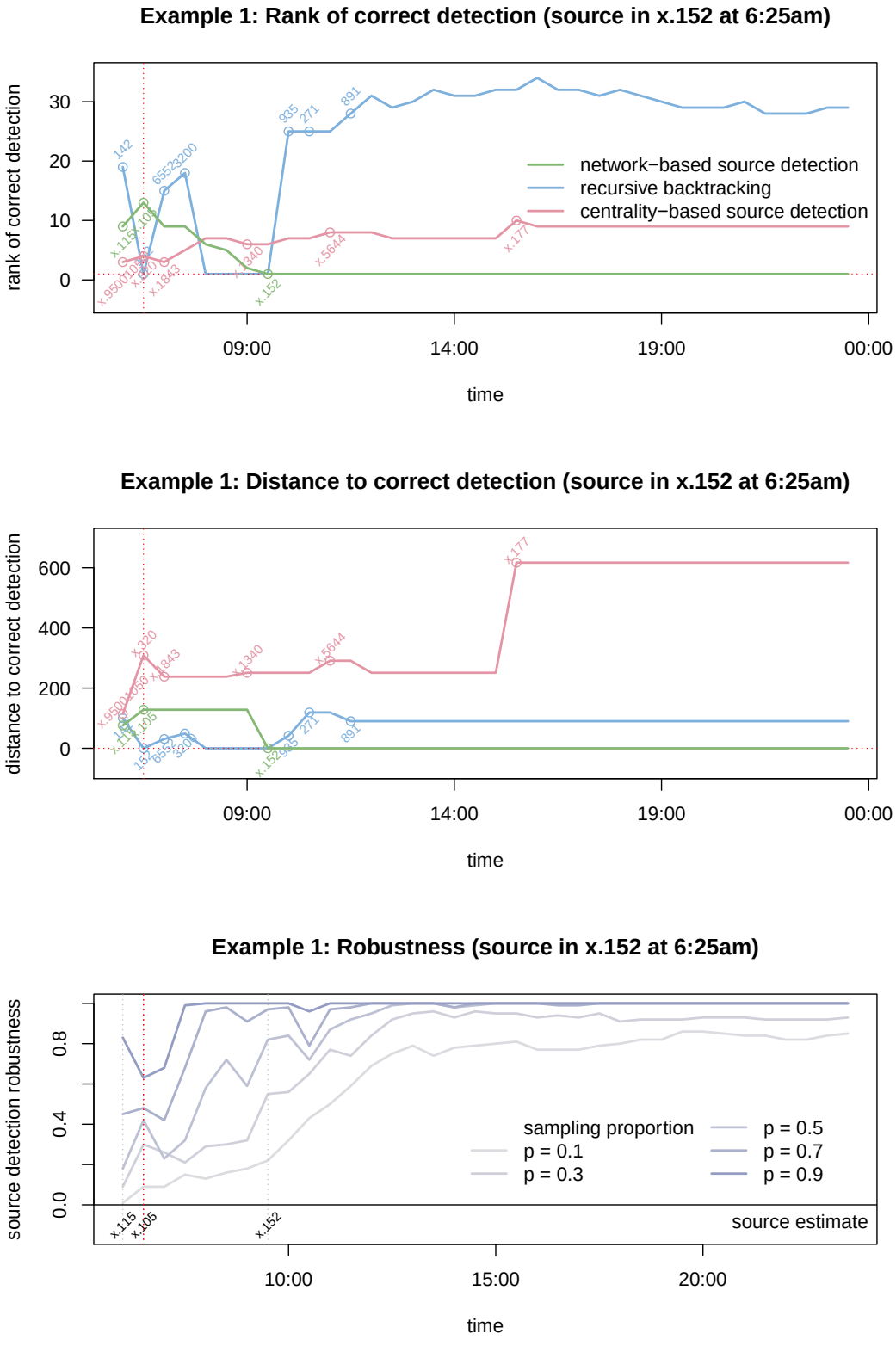


Figure 7: **Source detection in DB delay example 1 with delay source in station *x.152* at 6:25am.** Rank of correct detection, and distance to correct detection (travel time in minutes) for network-based source detection, recursive backtracking algorithm, and centrality-based source detection (Comin and da Fontoura Costa, 2011).

to be detected, in particular by the network-based source detection approach. This station is a node of high centrality and known to cause for a lot of disturbances in the German railway system. However, in the second example the network-based approach identifies the actual origin ($c_D = 36, c_B = 14,671$) even though a neighboring node has much higher centrality (*x.93244*; 9 minutes travel time; $c_D = 55, c_B = 30,189$).

5.2 Uncertainty Assessment via Subsampling for Network-Based Source Detection Estimates

The presented source detection methods result in estimates without declaring its certainty. In the following, we show the construction of an uncertainty assessment for the network-based source detection using a subsampling procedure. It is inspired by the idea of variation estimation with delete-d jackknife resampling, which is a linear approximation of bootstrap for estimate statistics that are not 'smooth' (Efron and Tibshirani, 1997).

We assume individual train information to be known and sample a subset of the trains with their punctuality information according to a sampling proportion $p \in [0, 1]$. The data subset is aggregated and the source detection method is applied. The resulting estimate is compared with the one obtained by applying source detection to the complete data set. By repeating the procedure for 100 times, we can calculate the relative ratio of runs, in which the same estimate is recovered as the one obtained with the complete data set. Since we use a subsampling technique without adjusting the robustness estimate accordingly, we underestimate the true estimation uncertainty. However, for a fixed subsampling proportion p, the results are comparable with on-going time. Therefore, the resulting value can be understood as an upper bound for the probability of estimate recovery.

Exemplarily, the procedure with the network-based source detection approach is applied to the first example of delay spreading on the German railway system using different sample proportions $p \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$ (see Figure 7). At 6:25am, the delays are introduced into the railway system. For detection estimate *x.105* between 6:30am until 9am, the robustness fluctuates for all sampling proportions considerably. For instance, for $p = 0.3$, the probability of estimate recovery ranges between 21% and 32%, whereas for $p = 0.7$ we obtain values between 42% and 98%. After detecting the correct origin in node *x.152* at 9:30am, the robustness for the estimate increases more steadily. For $p = 0.3$ probability of estimated recovery reaches 95% at 1pm, while for $p = 0.7$ such a high certainty is instantly reached at 9:30am. After that, we can observe a slight decrease of the estimated recovery rate. Note that the certainty values for $p = 0.7$ and $p = 0.9$ do not differ considerably, so that lower sampling proportions seem to be more informative. Using the uncertainty assesment on a specific case of application, where the true origin is not known, we suggest to select a fixed proportion for sampling depending on the presumed signal strength.

Altogether, these results seem to confirm the robustness of the network-based source detection that was observed in previous analyses. Additionally, this uncertainty assessment via subsampling allows for a quality evaluation of the obtained source estimate.

6 Conclusion and Discussion

As outlined in the introduction, source detection plays a key role in the assessment of various propagation processes. The network-based source detection method applied in this work was developed for infectious disease outbreaks. Still, with an enhanced version, this work illustrates the applicability of the method not only in the area of infectious diseases but also in the area of train delays. Additionally, we developed a backtracking approach, specifically designed for the source detection of delays in public transport. The approach yields good results in the train delay context, but we expect only limited ability to be extended to other areas of application. As a minimal data basis, both methods use the relative magnitude of train delays in a station, but in simple scenarios without noise the crude number of delays has been shown to be sufficient (see also Manitz, 2014).

Analyzing both methods in an extensive simulation study, the following findings can be summarized. As expected, source detection performance decreases over time, while a considerable effect of source node centrality cannot be verified. Further analysis showed a more pronounced effect in strongly centralized PTNs such as the Athens metro system (Manitz, 2014). In the German railway network, which shows typical network characteristics of general PTNs, the source detection approaches yield the best performances in contrast to results on the Göttingen bus network and the Athens metro system. However, the suggested recursive backtracking approach, which specifically considers the PTN propagation mechanism, is able to deal satisfactorily also with extreme network structures. Both methods are shown to be highly robust against various variants of delay spreading patterns. This includes the application of various delay management decisions. Further analyses exhibit that additional knowledge about train or passenger traffic when defining the network does improve the source detection performance only slightly (see supplementary material). Applying both methods to more realistic settings, considering background noise, the precision of detection is reduced in particular at the beginning of the spread, because first the source delay signal is indistinguishable from noise. The source detection performance for both methods improves over time, but eventually backtracking outperforms the network-based source detection in the artificial simulation setting, but showing larger variation in the source estimation.

In the application to delay spreading examples from the German railway operator *Deutsche Bahn*, both suggested methods show good performance with different strengths and weaknesses. Network-based source detection is a robust method for source detection, which steadily identifies the actual origin, but with some delay. Recursive backtracking seems to complement the

network-based source detection method by being a sensible method, which locates the area of origin very quickly, but the estimation dissolves after some time. Additionally, it was possible to construct an uncertainty assessment for source estimates by a subsampling strategy inspired by the idea of delete-d jackknife resampling. On the one hand, this analysis confirms the stability of the network-based source detection. On the other hand, we are able to compare the quality of the estimates over time. However, this is a simple assessment and we are working on a further development of this uncertainty measurement.

Beyond this, we were able to show the generalizability of the network-based approach to the propagation of train delays, but the backtracking method's applicability seems to be limited. Exemplarily, we analyzed the data from the 2011 *E. coli* outbreak in Germany (discussed in Manitz et al., 2014) with the newly developed backtracking approach. Results reveal that backtracking shows only very low detection performance on this example application (data not shown). Further analysis will be necessary to understand the full potentials of applicability of the suggested methods.

Moreover, our approaches are built for the detection of a single source. This assumption can be very restrictive. However, Zang et al. (2014) recently suggested a method that converts a multiple-source location problem into a number of single-source detection problems. This is based on the decomposition of the network graph into a number of subgraphs by applying simple community cluster algorithms. First simulations show promising results (see supplementary material), and it would be interesting to investigate the performance in an extensive simulation study.

Within the area of train delays, both methods can easily be applied to detect source delays on tracks as well. Assuming long time construction works or technical failures on tracks, also the source delays can be imposed on a track. Subsequently, a track as the cause of failure can be reconstructed with simple variants of the discussed methods.

Acknowledgments

We would like to thank the DB Mobility Logistics AG, *Verkehrsnetzentwicklung und Verkehrsmodelle (GSV)*, for the contribution of expert knowledge and making real data examples available. Special thanks go to Ingmar Schüle for the prosperous collaboration. This work was supported by the German Research Foundation, Research Training Group 1644 'Scaling Problems in Statistics' and the Simulation Science Center Clausthal/Göttingen.

Data Accessibility

Network data bases on Public Transportation Network, which is obtained from LinTim (Goerigk et al., 2015). This optimization software was utilized to generate the simulation data based on line plan, time table and delay propagation. The performance evaluation and statistical network analysis was conducted with the statistical software package R (R Core Team, 2014). A related R package covering the R code for this manuscript is in preparation. Due to a confidentiality agreement, we are not able to make the DB delay example data available.

References

Adar, E. and Adamic, L. (2005) Tracking information epidemics in blogspace. In *Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence*, 207–214. IEEE.

Albert, R., Albert, I. and Nakarado, G. (2004) Structural vulnerability of the north american power grid. *Physical Review E*, **69**.

Angeloudis, P. and Fisk, D. (2006) Large subway systems as complex networks. *Physica A: Statistical Mechanics and its Applications*, **367**, 553–558.

Brockmann, D. and Helbing, D. (2013) The hidden geometry of complex, network-driven contagion phenomena. *Science*, **342**, 1337–1342.

Büker, T. and Seybold, B. (2012) Stochastic modelling of delay propagation in large networks. *Journal of Rail Transport Planning & Management*, **2**, 34–50.

Comin, C. H. and da Fontoura Costa, L. (2011) Identifying the starting point of a spreading process in complex networks. *Physical Review E*, **84**.

Crucitti, P., Latora, V. and Marchiori, M. (2004) A topological analysis of the Italian electric power grid. *Physica A: Statistical Mechanics and its Applications*, **338**, 92–97.

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

Efron, B. and Tibshirani, R. J. (1997) *An introduction to the bootstrap*. New York [u.a.]: Chapman & Hall.

Eilers, P. H. and Marx, B. D. (1996) Flexible smoothing using B-splines and penalized likelihood. *Statistical Science*, **11**, 89–121.

von Ferber, C., Holovatch, T., Holovatch, Y. and Palchykov, V. (2009) Public transport networks: empirical analysis and modeling. *The European Physical Journal B*, **68**, 261–275.

Fioriti, V. and Chinnici, M. (2012) Predicting the sources of an outbreak with a spectral technique. *ArXiv e-prints*. URL`http://arxiv.org/pdf/1211.2333.pdf`.

Giannakis, M. and Louis, M. (2011) A multi-agent based framework for supply chain risk management. *Journal of Purchasing and Supply Management*, **17**, 23 – 31.

Goerigk, M., Harbering, J. and Schöbel, A. (2015) LinTim - Integrated Optimization in Public Transportation. URL`http://lintim.math.uni-goettingen.de/`.

Goverde, R. (1998) Optimal scheduling of connections in railway systems. *Paper Presented at the 8th WCTR, Antwerp, Belgium*.

Guimera, R., Mossa, S., Turtschi, A. and Amaral, L. A. N. (2005) The worldwide air transportation network: Anomalous centrality, community structure, and cities' global roles. *Proceedings of the National Academy of Sciences*, **102**, 7794–7799.

Kaluza, P., Kolzsch, A., Gastner, M. T. and Blasius, B. (2010) The complex network of global cargo ship movements. *Journal of The Royal Society Interface*, **7**, 1093–1103.

Kamada, T. and Kawai, S. (1989) An algorithm for drawing general undirected graphs. *Information Processing Letters*, **31**, 7–15.

Kolaczyk, E. D. (2009) *Statistical analysis of network data: methods and models*. Springer series in statistics. New York ; [London]: Springer.

Kuhr, D. (2013) Pünktlichkeit kennt keine Grenzen. Available online: http://www.sueddeutsche.de/reise/bahn-verspaetungen-puenktlichkeit-kennt-grenzen-1.1651879. Last access 03-04-2013.

Lappas, T., Terzi, E., Gunopulos, D. and Mannila, H. (2010) Finding effectors in social networks. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, 1059. ACM Press.

Li, W. and Cai, X. (2007) Empirical analysis of a scale-free railway network in china. *Physica A: Statistical Mechanics and its Applications*, **382**, 693–703.

Manitz, J. (2014) *Statistical Inference for Propagation Processes on Complex Networks*. Ph.D. thesis, Georg-August-University Göttingen.

Manitz, J., Kneib, T., Schlather, M., Helbing, D. and Brockmann, D. (2014) Origin detection during food-borne disease outbreaks - a case study of the 2011 EHEC/HUS outbreak in germany. *PLoS Currents Outbreaks*, **1**.

Meng, Y. (1991) *Bemessung von Pufferzeiten in Anschlüssen von Reisezügen.* 46. Veröffentlichungen des Verkehrswissenschaftlichen Institutes der Rheinischen-Wesfälischen Technischen Hochschule Aachen.

Nachtigall, K. (1998) Periodic network optimization and fixed interval timetables. *Deutsches Zentrum für Luft–und Raumfahrt, Institut für Flugführung, Braunschweig.*

Pinto, P. C., Thiran, P. and Vetterli, M. (2012) Locating the source of diffusion in large-scale networks. *Physical Review Letters*, **109**.

Plöchinger, S. and Jaschensky, W. (2013) So verspätet ist die Bahn in Ihrer Stadt, auf Ihrer Strecke. Available online: http://www.sueddeutsche.de/reise/verspaetungs-atlas-so-verspaetet-ist-die-bahn-in-ihrer-stadt-auf-ihrer-strecke-1.1651455. Last access 03-04-2013.

Prakash, B. A., Vreeken, J. and Faloutsos, C. (2012) Spotting culprits in epidemics: How many and which ones? In *Proceedings of the 2012 IEEE 12th International Conference on Data Mining (ICDM)*, 11–20. IEEE.

R Core Team (2014) *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria. URL`http://www.R-project.org/`.

Rigby, R. A. and Stasinopoulos, D. M. (2005) Generalized additive models for location, scale and shape (with discussion). *Applied Statistics*, **54**, 507–554.

Schachtebeck, M. (2010) *Delay Management in Public Transportation: Capacities, Robustness, and Integration.* Ph.D. thesis, Georg-August-University Göttingen.

Schachtebeck, M. and Schöbel, A. (2010) To wait or not to wait – and who goes first? Delay management with priority decisions. *Transportation Science*, **44**, 307–321.

Schöbel, A. (2001) A model for the delay management problem based on mixed-integer programming. *Electronic Notes in Theoretical Computer Science*, **50**.

— (2007) Integer programming approaches for solving the delay management problem. In *Algorithmic methods for railway optimization*, 145–170. Springer.

Schöbel, A. (2012) Line planning in public transportation: models and methods. *OR Spectrum*, **34**, 491–510.

Seaton, K. A. and Hackett, L. M. (2004) Stations, trains and small-world networks. *Physica A: Statistical Mechanics and its Applications*, **339**, 635–644.

Sen, P., Dasgupta, S., Chatterjee, A., Sreeram, P., Mukherjee, G. and Manna, S. (2003) Small-world properties of the indian railway network. *Physical Review E*, **67**.

Serafini, P. and Ukovich, W. (1989) A mathematical model for periodic scheduling problems. *SIAM Journal on Discrete Mathematics*, **2**, 550–581.

Shah, D. and Zaman, T. (2010) Detecting sources of computer viruses in networks: theory and experiment. In *Proceedings of the ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, 203–214. ACM.

— (2012) Rumor centrality: A universal source detector. In *Proceedings of the ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, 199–210. ACM.

Sienkiewicz, J. and Holyst, J. (2005) Statistical analysis of 22 public transport networks in Poland. *Physical Review E*, **72**.

Stasinopoulos, D. M. and Rigby, R. A. (2007) Generalized Additive Models for Location Scale and Shape (GAMLSS) in R. *Journal of Statistical Software*, **23**, 1–46.

Stevenson, M. D., Rossmo, D. K., Knell, R. J. and Le Comber, S. C. (2012) Geographic profiling as a novel spatial tool for targeting the control of invasive species. *Ecography*, **35**, 704–715.

Vansteenwegen, P. and Van Oudheusden, D. (2006) Developing railway timetables which guarantee a better service. *European Journal of Operational Research*, **173**, 337–350.

Yabuki, H., Ageishi, T. and Tomii, N. (2015) Mining the cause of delays in urban railways based on association rules. In *Proceedings of the 13 Conference on Advanced Systems in Public Transport*.

Yamamura, A., Koresawa, M., Adachi, S. and Tomii, N. (2013) Identification of causes of delays in urban railways. *Computers in Railways XIII: Computer System Design and Operation in the Railway and Other Transit Systems*, **127**, 403.

Zang, W., Zhang, P., Zhou, C. and Guo, L. (2014) Discovering Multiple Diffusion Source Nodes in Social Networks. *Procedia Computer Science*, **29**, 443–452.

Cheers!

First, I want to express my gratitude to Anita Schöbel. Even though you have a huge load of work, your door was always open and you took your time to discuss whatever I needed help with. Throughout my entire time working with you, you managed to create a positive and warm working environment, always keeping an eye on the needs of the people working with you. This work would not have been possible without your helping, encouraging and correcting words.

I want to thank the funding sources for the financial support. These are the European Union Seventh Framework Programme (FP7-PEOPLE- 2009-IRSES) under grant number 246647 with the New Zealand Government (project OptALI) and the Simulationswissenschaftliches Zentrum Clausthal-Göttingen (SWZ).

There was a group of people I had the great pleasure to work with. Through past and current research projects with E. Carrizosa, P. Gattermann, F. Kirchhoff, T. Kneib, M. Kolonko, J. Manitz, A. Ranade, M. Schmidt, A. Schöbel, and O. Sinnen I had the chance to learn many things which are both related to mathematics and how to work in a group.

Thanks a lot to Jule, Marc, and Marco for proof-reading this work. I did not at all expect that you would offer and then also accept the task of proof-reading without any hesitation! Thank you for that!

The warm and positive working environment was completed by the members of the working group: Alexander, Anita, Anja, Corinna, Jonas, Lisa, Marie, Marc, Marco Be., Marco Bo., Morten, Philine, Robert, Ruth, Sönke, Stephan, and Thorsten. One of the most important items on my "how I wish my working place to be"-list is definitely the people to work with, and this was a great one. Thank you for the great time spent together between work and leisure, coffee and drinks, table soccer and card games, chatting and relaxing.

Another source of support have always been my parents and my family. Vielen Dank für Eure kontinuierliche Unterstützung im Hintergrund, für das bedingungslose Zuhause-Gefühl und die vielen hilfreichen Gespräche.

Finally, thank you Sina! Thanks for the non-mathematical and mind-opening conversations all around, the full-hearted acceptance of my thinking mood, the moments of just being there, and, of course, the cover painting.

Curriculum Vitae

Jonas Harbering
Born the 13$^{\text{th}}$ of November, 1987 in Eckernförde, Germany

Since March 2013	Board Member of the Simulation Science Center Clausthal-Göttingen
Since Oct. 2012	PhD Candidate at the Institute for Numerical and Applied Mathematics, University of Göttingen, Germany
October 2012	Master of Sciences in Mathematics
Master Thesis	Global Approaches to Solving Min-Max-Location Problems in Cooperation with the University of Seville
Oct. 2011 – July 2012	Study Abroad, University of Seville, Spain
Oct. 2010 - Oct. 2012	Study of Mathematics (Master) with Minor in Economics, University of Göttingen, Germany
August 2010	Bachelor of Science in Mathematics
Bachelor Thesis	Train Delays in Public Transportation Networks in Cooperation with DSB S-tog (Danish State Railway, S-train Copenhagen)
Aug. 2009 – Feb. 2010	Study Abroad, University of Lund, Sweden
Oct. 2007 – Aug. 2010	Study of Mathematics (Bachelor) with Minor in Economics, University of Göttingen, Germany
June 2007	Allgemeine Hochschulreife (German School Leaving Examination), Humboldt-Gymnasium in Kiel