Line planning with user-optimal route choice

Marc Goerigk*[1] and Marie Schmidt†[2]

[1]Department of Management Science, Lancaster University, Lancaster, LA1 4YX, United Kingdom
[2]Rotterdam School of Management, Erasmus University Rotterdam, PO Box 1738, 3000 DR Rotterdam, The Netherlands

Abstract

We consider the problem of designing lines in a public transport system, where we include user-optimal route choice. The model we develop ensures that there is enough capacity present for every passenger to travel on a shortest route. We present two different integer programming formulations for this problem, and discuss exact solution approaches. To solve large-scale line planning instances, we also implemented a genetic solution algorithms.

We test our algorithms in computational experiments using randomly generated instances along realistic data, as well as a realistic instance modeling the German long-distance network. We examine the advantages and disadvantages of using such user-optimal solutions, and show that our algorithms sufficiently scale with instance size to be used for practical purposes.

Keywords: Transportation; line planning; public transport capacity; equilibrium constraints; railway optimization

1 Introduction

The acceptance of a public transportation system from a customer's point of view is highly dependent on the quality of service it provides, in particular in comparison with alternatives such as individual transport. If only poor connections are offered for an origin-destination-pair, passengers may decide to use alternative means of transportation or stay at home. Therefore, the quality of a public transportation system from the passengers' point of view is a key objective in designing public transportation systems, besides infrastructure constraints, operational limitations, and budget considerations.

However, taking passengers' behavior into account when solving optimization problems from public transportation planning complicates matters: in most public transportation systems, passengers cannot be steered by the system operator, but make their own travel choices, maximizing their benefit. The design of a public transportation system is hence a bilevel optimization problem: on the upper level the operator designs the system, on the lower level the passengers make travel choices in the system which optimize their individual objective function.

For the sake of tractability, mathematical programming approaches in passenger-oriented public transportation problems usually use strongly simplified demand models which avoid such bilevel modeling. Often, passenger routes are *fixed a priori*, i.e., they are assumed to be independent of the implemented transportation system. Models which do not presume fixed passenger routes normally operate under the implicit assumption that passengers can be *assigned* to routes by the operators. By design, (meta-)heuristic frameworks like iterative or genetic algorithms which split solution generation and solution evaluation, can handle more realistic demand models. However, these approaches can get stuck in suboptimal solutions.

*Corresponding author. Email: m.goerigk@lancaster.ac.uk
†Email: schmidt2@rsm.nl

In this paper we study the integration of passenger *route choice* in a simple *line planning* problem. The goal of line planning is to determine the routes which are served regularly by a vehicle (train, bus, tram, etc) and the frequencies of these services. Line planning is a so-called strategic public transportation problem, which is addressed early in the conventional public transportation process. The outcome, the so-called line concept, is then used as basis for further planning decisions like timetabling and vehicle scheduling.
We consider line planning as a bi-objective problem: Our goal is to find a line plan with low overall passenger travel time of the passengers and low costs for the operator.
In contrast to most previous work on this topic, we require our solution to be feasible in terms of transportation capacity, under the assumption that every passenger chooses the route which is 'best' for her/him. Hereby, we use a rather simplistic model for passengers' route choice behavior by assuming that all passengers aim at minimizing their travel times, composed of in-vehicle times and transfer times.
We model line planning with route choice (LPRC) as a bilevel optimization problem with a line planning problem on the upper level and passenger route choice problems on the lower level. We use two different techniques to transfer the bilevel optimization problem into a single-level (mixed-)integer linear problem ((M)IP). For both transformations, binary variables and 'big-M-constraints' need to be added, which lead to an increase in computation time compared to the problem formulation with route assignment instead of route choice. However, it turns out that in practical situations, most of the additional constraints are unnecessary. For this reason, we develop a constraint-generation approach, which iteratively adds the set of constraints to the formulation which are violated by the current solution.
In computational experiments, we find that these approaches are viable for smaller line planning problems with up to 10 stations; to solve real-world instances with up to 250 stations, we develop a genetic solution algorithm. Our results demonstrate that this algorithm is able to handle problems of such size, and produces reasonable trade-off solutions between passenger travel time and operator costs.

The remainder of this paper is organized as follows. In Section 2, we review the current literature on the topic and summarize our contribution. Our line planning models are formally introduced in Section 3, and different mathematical programming formulations are presented in Section 4. We discuss exact and heuristic solution approaches in Section 5, and present computational experiments in Section 6. We conclude the paper and point out further research questions in Section 7.

2 Related literature and contributions of this paper

In the following we review some relevant literature on routing in transportation (Section 2.1), line planning models (Section 2.2), integration of passenger routing in public transportation planning problems (Section 2.3), and, more general, integration of routing in network optimization problem (Section 2.4). In Section 2.5 we briefly explain the terminology used in this paper, before we place our contributions in the context of the literature.

2.1 Routing in transportation

There is a vast amount of research on how passengers behave in public transportation systems.
Research on *route choice* often follows a behavioral approach to investigate which aspects are decisive when choosing a route in a transportation network with the aim to build quantitative models that predict individual's route choice as good as possible.
Traffic assignment and *transit assignment* models, on the other hand, often take a system perspective and model route choice behavior of the passengers as a whole, normally focussing on congestion aspects. Hereby, the term traffic is mostly used when referring to individual transportation, for public transportation, normally the term transit assignment is used.

These research directions have brought forth a variety of sophisticated demand models, including among others discrete-choice assignment and equilibrium models, see, e.g., [OW11, BS12, WN13, Pat15].
In *route planning*, the focus is more on computationally efficient methods to guide passengers through a transportation network (see [DSSW09] for a survey). Finding routes through train networks (see [MHSWZ07, GSS+13b] is computationally more challenging than for street networks, where a road hierarchy can be used.
Line planning models are usually based on (mixed-)integer programming formulations; due to their complexity, they require simple models of passenger behavior that may be included as side constraints. The purpose of this paper is to improve the realism of passenger behavior by including an egoistic route choice, instead of a system-optimal passenger routing. The way the quality of a path is assessed by a passenger, however, is similar to previous papers on the topic.

2.2 Line planning models

There is a large amount of work on line planning in the literature, comprising a variety of models. Models differ by the decisions covered (determination of train routes, frequency setting, or both), infrastructure and operational aspects taken into account, objective functions, and the way passengers' decisions are taken into account in the decision process. For a more extensive overview on line planning concepts and models, see [Sch12b].

While some line planning literature considers only the determination of train routes [GYW06, SS15a, Sch14, Har16] or only frequency setting [CF95, GSS04, dMI06], most commonly both problems are addressed together. In fact, there are various reasons why frequencies should be included in the line planning process:
Firstly, frequencies should be high enough to provide enough transport capacity. Secondly, frequencies have an impact on the transfer times and should thus be of interest when considering passenger travel time and in particular when routing passengers. However, incorporation of frequency-based transfer times leads to non-linear optimization models. For this reason, most literature in this area estimates transfer times by a transfer penalty, see e.g., [SS06, NJ08, BGP07, BK12, Sch14]. For models which do take frequency-based transfer times into account we refer the interested reader to [CF95, GSS04, dMI06] and references therein.

Many line planning papers consider upper bounds on edge frequencies, see, e.g., [GvHK06, GvHK04, SS06, GYW06]. Operational costs are taken into account as simple frequency-based costs [GYW06, SS15a, Sch14, SS06], sum of fixed and frequency-based costs [BGP07, BK12], or using more sophisticated models, e.g., based on rolling stock utilization [GvHK06, GvHK04].

With respect to objective functions, line planning models can be roughly classified into models with cost-oriented or with passenger-oriented objective functions. Focusing on the costs that arise from establishing the lines rather than on passengers' convenience, many *cost-oriented* models use a covering approach, assuming pre-estimated minimal frequencies of service on the tracks to model passengers' demand [GvHK06, GvHK04]
In contrast to cost-oriented approaches, *passenger-oriented* approaches focus on the optimization of passenger-related criteria. One example is the maximization of the number of direct travelers [BKZ97, Bus98]. The objective of minimizing travel time has been treated, e.g., in [Sch05, SS06, SS12, Sch12a]. In [BGP07, BGP08, PB06], a weighted sum of operational costs and travel time is minimized. When using travel time as an objective function, fixed passenger routes, as they are used, e.g., in [BKZ97, CvDZ98, GvHK04], are not suitable because travel time does not only depend on track lengths, but also on transfer times. In the following, we discuss details on the integration of routing aspects.

2.3 Integration of passenger demand in public transportation

While, for the sake of simplicity, many older models in public transportation optimization consider demand and even route choice as fixed (see the survey [FMSR13]), there are recent approaches

to integrate passenger route choice in typical optimization problems from public transportation like line planning [GYW06, SS06, NJ08, BGP07, BK12, SS15a, Sch14, Har16], timetabling [SG13, SS15b, Sch14, HBK15], delay management [DHSS12, Sch13, Sch14], and rolling stock rescheduling [KMN14, vKM16].
An early approach to frequency setting with integrated passenger routing is described in [CF95]. The problem is formulated using a bilevel model and solved via a subgradient approach. Some other papers, see, e.g., [GSS04] use heuristics to tackle bilevel modelling approaches.
In order to model the described problems with integrated passenger route choice as MIPs, it is often implicitly assumed that the number of passengers per origin-destination pair per time is fixed and that

(a) there are no capacity restrictions on the vehicles, passengers choose travel-time minimal routes or routes with minimal transfers, and the operator minimizes overall travel time (*uncapacitated route choice*), see, e.g., [GYW06, SS15a, Sch14, Har16, SG13, SS15b, DHSS12, Sch13, Sch14, HBK15], or

(b) passengers can be steered by the public transportation operator (*route assignment*), see, e.g., the publications [SS06, NJ08, BGP07, BK12, Sch14] on line planning.

Using these assumptions, instead of using a bilevel formulation, the public transportation problem with integrated route choice can be expressed as a MIP with flow or path constraints modelling passengers' route choice.
However, public transportation systems do not (always) comply with assumption (a). In particular in line planning problems involving frequency setting, frequencies are often chosen based on capacity considerations. In this case, assumption (b) allows to incorporate capacity restrictions. However, using route assignment as described in (b) can lead to unrealistic and even infeasible situations, as we will demonstrate in Example 3.2.
Besides travel times and transfers, also other factors like congestion and ticket costs play a role in route choice. More sophisticated demand models are taken into account in metaheuristic approaches like genetic or evolutionary algorithms [FM06, BCCP09, SMFZ10, SK12] or iterative approaches as suggested in [Sch14, KMN14, vKM16]. However, these approaches may get stuck in suboptimal solutions.
Fewer attempts have been made to integrate mode choice into transportation optimization, see e.g., [LMOS05, LMMO07, CR11].

2.4 Integration of network design and routing

The public transportation planning problems reviewed in Section 2.3 can be modeled as network optimization problems with integrated routing decisions. Similar problems can be found in many other applications, such as road network design [Mig95], toll setting [BLMS01], transportation of hazardous materials [KV04], location of speed-up subnetworks [SS14], network interdiction [Woo11], shipper-carrier interactions [LSC$^+$14] and telecommunication [AFTÜ09], see also [CMS07] for an overview. All of these problems have in common that they can be modeled as bilevel problems where a planner makes network design decisions on the upper level, and routing decisions are made on the lower level. In some applications the routing decisions are centralized (route assignment), e.g., when a shipper prescribes transportation routes for the carriers, in other applications they are decentralized (route choice), e.g., in road network design, where each car driver makes an individual decision on which route to take.
Furthermore, the contrast between route assignment and route choice is closely related to game theoretical concepts. In the context of *routing games*, the *price of anarchy*, comparing *equilibrium* solutions obtained by *selfish routing* to system-optima is an interesting object of study (see, e.g., [Rou05]). In *Stackelberg games* a hierarchical setting is assumed: first the *leader* makes some *upper-level* decisions, anticipating the followers reaction to these decisions, then a so-called *Stackelberg equilibrium* for the followers is calculated (see, e.g., [BS02, KLO97, Rou04]). In particular if the followers' decisions do not interdepend mutually, i.e., every follower optimizes his individual

objective function in the framework provided by the leader's decisions, the problems are often formulated and solved as *bilevel programs*.

2.5 Terminology and contributions of this paper

In this paper we distinguish between line planning problems in which the line plan needs to be feasible under the assumption that each passenger chooses the route which appears best to her / him, as opposed to approaches which may assign passengers to routes which are not optimal with respect to the passenger's objective (as described under header (b) in Section 2.3). We refer to the first problem as *Line planning with route choice*, and to the latter as *line planning with route assignment*.
We study line planning with route choice (LPRC) as introduced in [Sch14] under the name *capacitated line planning with routing*. We start with the formulation of the problem as a bilevel optimization problem. We then transfer the bilevel problem into a single-level IP using two different approaches: the first one is based on the dualization of the inner-level routing problem, the second one is based on implicit shortest-path constraints and was first described in [Sch14].
In both cases, the resulting IP model can be read as a line planning with route assignment (LPRA) model with additional constraints and variables. In particular because some of these variables are binary, and the constraints contain 'big-M'-values, the resulting models are harder to solve than LPRA when using standard solvers. However, based on the observation that most of the added constraints may not be necessary, we develop a constraint-generation method which starts with solving LPRA, and iteratively detects violated constraints and adds them to the model. We conduct extensive experiments in which we compare both models for LPRC. Furthermore, we compare the obtained solutions to LPRA solutions to investigate how often LPRA indeed leads to solutions which violate the assumption that passengers travel on shortest paths, and by how much these are violated. This allows us to comment on the trade-off between solution quality and solution time when comparing LPRC and LPRA. To solve larger problems, we propose a multi-objective genetic algorithm, which is able to find trade-off solutions between travel time and line concept costs for real-world sized instances in reasonable time.

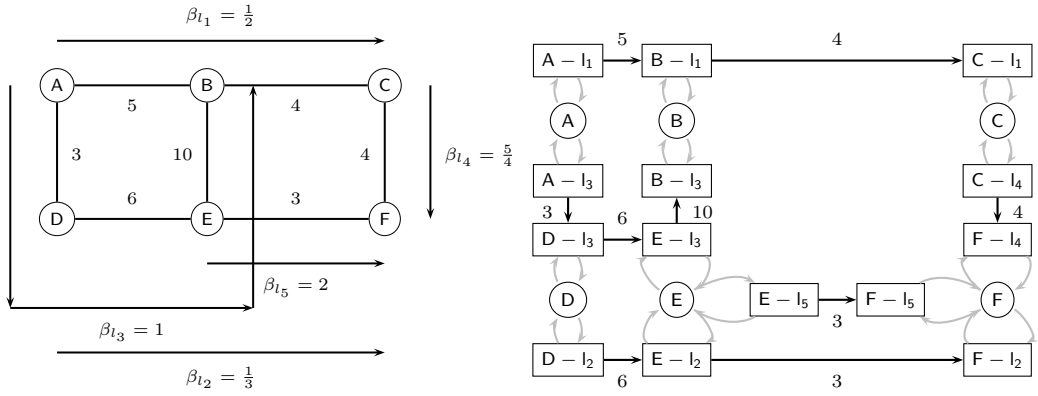
For our study we use a simple version of the line planning problem, as described in the following Section 3. However, the techniques we apply to transfer the bilevel problem to an IP are independent of modeling decisions like the choice of objective function and of the upper level constraints, i.e., infrastructure and most operational line planning constraints. They could hence be adapted to many other line planning models, other capacitated public transportation problems with integrated route choice, or in general network optimization problems with integrated routing.

3 Line planning with route choice

As discussed in Section 2, there is a wide variety of models for line planning. In this paper we consider the following simple base model:

We make use of a network which provides information about the stations and the tracks connecting the stations, the *public transportation network (PTN)* $\mathcal{G} = (S, E)$. In railway transportation, the vertices S represent train stations, the (undirected) edges E represent the physical connections between the vertices, i.e., the tracks. Edge lengths $L_e \in \mathbb{N}_0$ denote geographic distances between stations.
A *line* l is a directed path in the public transportation network $\mathcal{G}$ given as a sequence of stations and edges $(s_1, e_1, s_2, e_2, \ldots, s_k)$. To keep notation short, we sometimes describe a line stating only the corresponding sequence of stations $(s_1, s_2, \ldots, s_k)$ or the corresponding sequence of edges $(e_1, e_2, \ldots, e_{k-1})$. We assume that lines are chosen from a predefined *line pool* $\mathcal{L}$.

The task of line planning implies choosing a subset of lines $\mathcal{L}' \subset \mathcal{L}$ of lines and a frequency $f_l \in \mathbb{N}$ for every $l \in \mathcal{L}'$. The vector of frequencies $F := (f_l)_{l \in \mathcal{L}}$, where we set $f_l := 0$ for every $l \notin \mathcal{L}'$, is called *line concept*.



(a) PTN and lines

(b) The CGN. Black arcs are travel arcs with length as indicated, grey arcs are transfer arcs with length $\frac{p}{2}$.

Figure 1: Example for the construction of a CGN.

Every line $l \in \mathcal{L}$ is assigned a *cost* b_l that arises for every train of line l which is operated in the given time period. The *total cost for trains of line* l is then calculated as $f_l \cdot b_l$, the cost of a line concept F is $\sum_{l \in \mathcal{L}} f_l b_l$. Furthermore, for every $l \in \mathcal{L}$ we associate a monotonously increasing *line speed function* $\alpha_l : \mathbb{R} \to \mathbb{R}$ which transforms the length L_e of an edge e to the driving time $\alpha_l(L_e)$ of line l on e. For the sake of simplicity, for the examples and experiments in this paper we assume that α_l is defined as $\alpha_l(L_e) := \frac{L_e}{\beta_l}$ for a *line speed factor* β_l. However, our models are also able to include more complex relationships.
The goal of line planning is to find a line concept which leads to low costs on the one hand, and low passenger travel time on the other hand.

We assume that the number of passengers wanting to travel by train between two stations is fixed. We denote by $\mathcal{OD} = \{(u_1, v_1), \ldots, (u_K, v_K)\} \subset S \times S$ the origin-destination (OD) pairs, i.e., the set of station pairs (u_k, v_k) with positive travel demand w_k. The set of indices $\{1, \ldots, K\}$ is denoted as $\mathcal{K}$. The travel time of a passenger on a route from origin to destination is computed as the sum of travel times, plus the sum of transfer times, which are estimated by a fixed penalty p for every transfer. Our model readily extends to penalties $p_s^{ll'}$ that depend on the station and the two lines that are involved.

In order to obtain a better representation of passengers' routes we make use of a *change & go network (CGN)* $\mathcal{N} = (V, A)$. This concept was previously defined in [SS06, Sch14]; in this paper we introduce an improved version which has the advantage that the number of arcs is only linear in the number of lines and stations, instead of quadratic.
The node set V of $\mathcal{N}$ consists of a set of *station nodes* V_{station} (containing one node for each station $s \in S$) and travel nodes V_{travel} (containing one node for each station $s \in S$ and each line $l \in \mathcal{L}$ that visits the station). The arc set A of the CGN consists of *travel arcs* A_{travel} which connect travel nodes along each line, and transfer arcs A_{transfer}, which connect each station node to all corresponding travel nodes. For each $a \in A_{\text{travel}}$ we define its travel time as $c_a := \alpha_l(L_e)$ for the corresponding track e of the PTN, for each transfer arc we set $c_a := \frac{p}{2}$. We denote by $l(a)$ the line associated to $a \in A$. An example for this construction is given in Figure 1.
Now we can define a passenger's *route* as a path in the CGN from the station node corresponding to the passengers' origin to the station node corresponding to his destination. Note that a route describes not only the physical path a passenger takes (which corresponds to the chosen path in the PTN) but also contains information on which line is taken on which part of the physical path. The passengers' travel time, including in-vehicle time and transfer time, can then be computed

as the sum over all arc lengths on the passenger's route, except for the first and last arc (because transfer times should not be added for boarding the first train and alighting from the last).
Naturally, the routes that can be taken by the passengers depend on the line concept. In particular, only lines l that are established, i.e., lines with $f_l > 0$ can be used by passengers. We call a route r *compatible* with a line concept F if it is contained in the routing network $\mathcal{N}(F) = (V(F), A(F))$ from which all travel nodes corresponding to lines with $f_l = 0$ and all incident arcs are removed.
Passengers belonging to the same OD-pair do not necessarily need to travel on the same route. For $k \in \mathcal{K}$, we call $\mathcal{R}_k := \{r : \text{ route } r \text{ is used by at least one passenger of OD-pair } (u_k, v_k)\}$ a *routing for OD-pair* k and denote by w_k^r the number of passengers of OD-pair k using r. Of course, $\sum_{r\in\mathcal{R}_k} w_k^r = w_k$ must hold. A *routing for* $\mathcal{OD}$ is defined as $\mathcal{R} := \bigcup_{k\in\mathcal{K}} \mathcal{R}_k$. Let us denote by Cap the capacity of one vehicle. A routing is *compatible* with a line concept F if each route is compatible with F and the capacity of the line concept is sufficient to transport all passengers, i.e., for all travel arcs $a \in A_{\text{travel}}$ is holds that

$$\sum_{k\in\mathcal{K}} \sum_{r\in\mathcal{R}_k : r\ni a} w_k^r \leq f_{l(a)}\text{Cap}. \tag{1}$$

It is called a *shortest-path routing* for F, if for every $k \in \mathcal{K}$, every $r \in \mathcal{R}_k$ is a shortest path in the routing network $\mathcal{N}(F)$.
Now the (biobjective) line planning problem with route choice can be defined as follows:

Definition 3.1. *An instance* $(\mathcal{G}, \mathcal{L}, \mathcal{OD}, Cap)$ *of the* line planning problem with route choice (LPRC) *consists of a PTN* $\mathcal{G}$*, a line pool* $\mathcal{L}$*, a set of OD-pairs* $\mathcal{OD}$*, and a restriction on the capacity of the trains Cap.*
The task is to choose a line concept F *and a compatible shortest path routing* $\mathcal{R}$ *in* $\mathcal{N}(F)$ *such that total costs of the line concept and total travel time of the passengers are minimized.*

In comparison, we call the problem of finding a line concept together with a compatible routing (which is not necessarily a shortest-path routing) which minimizes total costs and total travel time line planning with route assignment (LPRA) problem.

The following example illustrates the differences between LPRC and LPRA.

Example 3.2. *We are given a PTN with four stations as shown in Figure 2. There are three lines with line speed factor* $\beta_l = 1$*: line* l_1 *from* s_1 *via* s_2 *to* s_4*, line* l_2 *from* s_2 *to* s_4*, and line* l_3 *from* s_1 *via* s_3 *to* s_4*. The line costs are* $b_{l_1} = 3$*,* $b_{l_2} = 1$*, and* $b_{l_3} = 2$*. We assume a train capacity of* $Cap = 100$*. There are* 50 *passengers travelling from* s_2 *to* s_4 *and from* s_3 *to* s_4*, and* 100 *from* s_1 *to* s_4*. We now look at the best solution to LPRC and LPRA with respect to travel time, whose total cost does not exceed* 5*.*

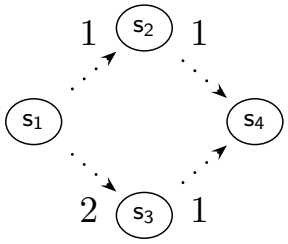


Figure 2: PTN of an instance where solutions to LPRA and LPRC are different.

We note that for the passengers with origin s_2 *and* s_3 *there is only one possible physical path in the network, while passengers with origin* s_1 *could reach their destination via* s_2 *or* s_3*. It is easy to check that in this situation, the travel-time optimal combination of frequency assignment and compatible routing is to install lines* l_1 *and* l_3 *with frequency* 1*. In this case, passengers from* s_1 *to* s_4 *have to split,* 50 *of them can take the shorter route via* s_2*, while the other* 50 *have to take the*

longer route via s_3. This is thus the optimal solution to LPRA with objective value 350. *However, this solution would only be feasible in a realistic situation, if passengers could really be* assigned *to the route via s_3. When passengers choose their routes freely, it is hard to imagine that* 50 *of them would voluntarily embark on a slower route. Most likely, all* 100 *would choose the route via s_2, making it impossible (or: very inconvenient) for the passengers with origin s_2 to board the train. The only combination of frequency assignment and compatible* shortest-path *routing is to install line l_2 with frequency* 1 *and line l_3 with frequency* 2. *In this case, passengers from s_1 have only one route choice option, namely the route via s_3 and will thus embark on this route willingly. However, this comes at the cost of a higher overall travel time of* 400.

It has been shown in [Sch14] that finding a solution for a given cost budget to LPRA and LPRC is NP-hard, even if there is only one OD-pair and there are no transfer penalties. In general, it may be that LPRA is feasible for some cost budget, but LPRC is not.

4 Mathematical programming formulation

4.1 A bilevel formulation for LPRC

We now formulate LPRC as a bilevel, biobjective optimization problem.

$$
\begin{aligned}
(LPRC-bil)\quad & \min \sum_{k\in\mathcal{K}}\sum_{a\in A} c_a x_a^k, && && (2)\\
& \min \sum_{l\in\mathcal{L}} f_l b_l, && && (3)\\
& \quad s.t. \sum_{k\in\mathcal{K}} x_a^k \le f_{l(a)}\mathrm{Cap} && \forall\ a\in A, && (4)\\
& \quad x_a^k \text{ is an optimal solution to } (RP_k) && \forall\ k\in\mathcal{K} && (5)\\
& \quad f_l \in \mathbb{N} && \forall\ l\in\mathcal{L}, && (6)\\
& \quad x_a^k \in \mathbb{N} && \forall\ k\in\mathcal{K},\ a\in A, && (7)\\
& && && (8)
\end{aligned}
$$

where the lower level *routing problem* (RP_k) for OD-pair $k\in\mathcal{K}$ is defined as

$$
\begin{aligned}
(RP_k)\quad & \min \sum_{a\in A} c_a \hat{x}_a^k && && (9)\\
& \quad s.t. \sum_{a\in\delta^-(u_k)} \hat{x}_a^k = w_k && && (10)\\
& \sum_{a\in\delta^+(v)} \hat{x}_a^k - \sum_{a\in\delta^-(v)} \hat{x}_a^k = 0 && \forall\ v\in V\setminus\{u_k,v_k\}, && (11)\\
& \sum_{a\in\delta^+(v_k)} \hat{x}_a^k = w_k && && (12)\\
& \hat{x}_a^k \le f_{l(a)}\cdot w_k && \forall\ a\in A, && (13)\\
& \hat{x}_a^k \in \mathbb{N} && \forall\ a\in A. && (14)
\end{aligned}
$$

Here we denote by $\delta^+(v)$ and $\delta^-(v)$ the set of incoming and outgoing arcs of node $v\in V$, respectively. The upper level of the problem corresponds to the operator's line planning problem. We use non-negative integer variables f_l for the frequencies and non-negative integer flow variables x_a^k to model the passenger routes. More precisely, x_a^k denotes the number of passengers of OD-pair k using arc a of the CGN.
The operator has to make sure that the transportation capacity on all travel edges is sufficient to transport the passengers wanting to travel on them, as described in constraints (4).
The lower level routing problems (RP_k) model the route choice of the passengers. Here, the variables f_l from the upper level are fixed. Flow variables $\hat{x}_a^k$ for all arcs a model the route choice

of the passengers of OD-pair k. Note that since we assume that passengers travel on travel-time minimal routes and that arc lengths represent travel times and are hence positive, we can assume that $\hat{x}_a^k \in [0, w_k]$ for all a.
Note that (RP_k) is the flow formulation of a shortest path problem in the CGN, with one additional constraint (13). If $f_l = 0$ for a line l, constraint (13) reads $\hat{x}_a^k \leq 0$ for all travel arcs a belonging to that line. On the other hand, if $f_l \geq 1$, constraint (13) is trivially fulfilled by any optimal solution to (RP_k).
Hence, (RP_k) is a shortest path problem in $\mathcal{N}(F)$. That means that an optimal solution to (RP_k) is a travel-time minimal routing for OD-pair k.

4.2 An integer program for LPRA

When we do not require that each passenger travels on a shortest route (constraints (5)), but model passenger flows (only) via flow constraints, we obtain an IP formulation of line planning with route assignment, as studied, e.g., in [Sch14] and similarly in [SS06, BGP07].

$$(LPRA) \quad \min \sum_{k\in\mathcal{K}} \sum_{a\in A} c_a x_a^k, \tag{15}$$

$$\min \sum_{l\in\mathcal{L}} f_l b_l, \tag{16}$$

$$s.t. \sum_{k\in\mathcal{K}} x_a^k \leq f_{l(a)}\mathrm{Cap} \qquad \forall\, a \in A, \tag{17}$$

$$\sum_{a\in\delta^-(u_k)} x_a^k = w_k \qquad \forall\, k \in \mathcal{K} \tag{18}$$

$$\sum_{a\in\delta^+(v)} x_a^k - \sum_{a\in\delta^-(v)} x_a^k = 0 \qquad \forall\, k \in \mathcal{K},\ v \in V \setminus \{u_k, v_k\}, \tag{19}$$

$$\sum_{a\in\delta^+(v_k)} x_a^k = w_k \qquad \forall\, k \in \mathcal{K} \tag{20}$$

$$f_l \in \mathbb{N} \qquad \forall\, l \in \mathcal{L}, \tag{21}$$

$$x_a^k \in \mathbb{N} \qquad \forall\, k \in \mathcal{K},\ a \in A, \tag{22}$$

Note that more passenger routes are feasible in LPRA than in LPRC-bil, as we neglect the constraint that passenger routes need to be shortest paths in the latter. Therefore, LPRA is a relaxation of LPRC-bil.
Indeed, as we have seen in Section 3, there are instances in which LPRA achieves better objective values than LPRC-bil – however, this comes at the disadvantage that some passengers are assumed to travel on paths which are not shortest with respect to travel time in the routing network.

4.3 Integer programming formulations for line planning with route choice

In the following, we describe to different integer programming formulations that linearize the bilevel problem LPRC-bil.

4.3.1 First model

In our first approach, we reconsider the lower lever routing problem (RP_k). We introduce a new binary variable y_a that denotes the availability of an arc, i.e., it is equal to one iff $f_{l(a)} > 0$. As noted, we may relax the variables $\hat{x}_a^k$. The dual of (RP_k) is hence given as

$$(D - RP_k) \quad \max \pi_{v_k}^k - \pi_{u_k}^k - \sum_{a\in A} \alpha_a y_a, \tag{23}$$

$$s.t. \ \ \pi_j^k - \pi_i^k - \alpha_{ij} \leq w_k c_a \qquad \forall\, a = (i,j) \in A, \tag{24}$$

$$\pi_i^k \geq 0 \qquad \forall\, i \in V, \tag{25}$$

$$\alpha_a^k \geq 0 \qquad \forall\ a \in A. \tag{26}$$

Using strong duality, we can therefore ensure that all passengers travel on shortest paths with the help of constraints (24-26) and constraints of the form

$$\sum_{a \in A} c_a x_a^k = \pi_{v_k}^k - \pi_{u_k}^k - \sum_{a \in A} \alpha_a^k y_a \qquad \forall\ k \in \mathcal{K}$$

in the upper level problem. However, the product $\alpha_a^k y_a$ is not linear anymore, and needs to be linearized by introducing new variables $\beta_a^k = \alpha_a^k y_a$. Overall, this linearization of problem LPRC-bil can be written as:

$$(LPRC-1) \quad \min \sum_{k \in \mathcal{K}} \sum_{a \in A} c_a x_a^k, \tag{27}$$

$$\min \sum_{l \in \mathcal{L}} f_l b_l, \tag{28}$$

$$s.t. \sum_{a \in \delta^-(u_k)} x_a^k = w_k \qquad \forall\ k \in \mathcal{K} \tag{29}$$

$$\sum_{a \in \delta^+(v)} x_a^k - \sum_{a \in \delta^-(v)} x_a^k = 0 \qquad \forall\ k \in \mathcal{K},\ v \in V \setminus \{u_k, v_k\}, \tag{30}$$

$$\sum_{a \in \delta^+(v_k)} x_a^k = w_k \qquad \forall\ k \in \mathcal{K}, \tag{31}$$

$$\sum_{k \in \mathcal{K}} x_a^k \leq f_{l(a)} \mathrm{Cap} \qquad \forall\ a \in A, \tag{32}$$

$$y_a \leq f_{l(a)} \leq M_1^l y_a \qquad \forall\ a \in A, \tag{33}$$

$$\sum_{a \in A} c_a x_a^k = \pi_{v_k}^k - \pi_{u_k}^k - \sum_{a \in A} \beta_a^k \qquad \forall\ k \in \mathcal{K}, \tag{34}$$

$$\pi_j^k - \pi_i^k - \alpha_{ij}^k \leq w_k c_a \qquad \forall\ k \in \mathcal{K},\ a \in A, \tag{35}$$

$$\beta_a^k \leq M_2^{ka} y_a \qquad \forall\ a \in A,\ k \in \mathcal{K} \tag{36}$$

$$\alpha_a^k - M_2^{ka}(1 - y_a) \leq \beta_a^k \leq \alpha_a^k + M_2^{ka}(1 - y_a) \qquad \forall\ a \in A,\ k \in \mathcal{K} \tag{37}$$

$$f_l \in \mathbb{N} \qquad \forall\ l \in \mathcal{L}, \tag{38}$$

$$x_a^k \in \mathbb{N} \qquad \forall\ k \in \mathcal{K},\ a \in A, \tag{39}$$

$$y_a \in \{0, 1\} \qquad \forall\ a \in A, \tag{40}$$

$$\pi_i^k \geq 0 \qquad \forall\ k \in \mathcal{K},\ i \in V, \tag{41}$$

$$\alpha_a^k \geq 0 \qquad \forall\ k \in \mathcal{K},\ a \in A, \tag{42}$$

$$\beta_a^k \geq 0 \qquad \forall\ k \in \mathcal{K},\ a \in A. \tag{43}$$

Here, M_1^l and M_2^{ka} are large enough constants (it suffices to set $M_1^l \geq \lfloor B/b_l \rfloor$ and $M_2^{ka} \geq w_k \sum_{a \in A} c_a$). The constraints (29) to (32) are the same as in LPRA, i.e., all passengers are assigned routes in accordance with line capacities. The additional constraints (33)–(37) ensure that only shortest paths can be assigned to passengers, which is modeled via strong duality in (34) and the dual constraints (35), as well as the linearization of the product $\alpha_a^k y_a$ in (33)-(37).

4.3.2 Second model

The second IP model for LPRC is obtained from LPRA by adding constraints (50-59) which make sure that passengers can only travel on shortest paths. To this end, we introduce binary variables z_a^k which indicate whether arc $a = (i, j)$ lies on a shortest path from node i to v^k. To determine the values of z, we introduce auxiliary variables y_l for each line l and r_i^k for each OD-pair k and each node i. The variable y_l indicates whether line l is established in the line concept F and hence determines whether arcs $a = (i, j) \in l$ have to be taken into account when determining the

shortest paths from i to a destination v^k in the routing network. The node potential r_i^k indicates the shortest path distance from node i to destination v^k in the routing network.

$$(LPRC-2) \quad \min \quad \sum_{k\in\mathcal{K}}\sum_{a\in A} c_a x_a^k, \tag{44}$$

$$\min \quad \sum_{l\in\mathcal{L}} f_l b_l, \tag{45}$$

$$s.t. \quad \sum_{a\in\delta^-(u_k)} x_a^k = w_k \qquad \forall\, k\in\mathcal{K}, \tag{46}$$

$$\sum_{a\in\delta^+(v)} x_a^k - \sum_{a\in\delta^-(v)} x_a^k = 0 \qquad \forall\, k\in\mathcal{K},\ v\in V\setminus\{u_k,v_k\}, \tag{47}$$

$$\sum_{a\in\delta^+(v_k)} x_a^k = w_k \qquad \forall\, k\in\mathcal{K}, \tag{48}$$

$$\sum_{k\in\mathcal{K}} x_a^k \le f_{l(a)}\mathrm{Cap} \qquad \forall\, a\in A, \tag{49}$$

$$y_l \le f_l \qquad \forall\, l\in\mathcal{L}, \tag{50}$$

$$M_3\cdot y_l \ge f_l \qquad \forall\, l\in\mathcal{L}, \tag{51}$$

$$z_a^k \le y_{l(a)} \qquad \forall\, a\in A,\ k\in\mathcal{K}, \tag{52}$$

$$r_j^k + c_{ij} - r_i^k + z_{ij}^k + M_4^{ij}(1-y_{l(i,j)}) \ge 1 \qquad \forall\,(i,j)\in A,\ k\in\mathcal{K}, \tag{53}$$

$$r_j^k + c_{ij} - r_i^k - M_4^{ij}\cdot(1-z_{ij}^k) \le 0 \qquad \forall\,(i,j)\in A,\ k\in\mathcal{K}, \tag{54}$$

$$r_{v_k}^k = 0 \qquad \forall\, k\in\mathcal{K}, \tag{55}$$

$$x_a^k \le z_a^k\cdot w_k \qquad \forall\, a\in A,\ k\in\mathcal{K}, \tag{56}$$

$$r_i^k \in \mathbb{N} \qquad \forall\, k\in\mathcal{K},\ i\in V, \tag{57}$$

$$x_a^k\in\mathbb{N}, z_a^k\in\{0,1\} \qquad \forall\, k\in\mathcal{K},\ a\in A, \tag{58}$$

$$f_l\in\mathbb{N}, y_l\in\{0,1\} \qquad \forall\, l\in\mathcal{L}. \tag{59}$$

In constraints (50-51) y_l is set to 1 if $f_l > 0$ and to 0 otherwise for M_3 big enough. Since in the worst case a line needs to transport all passengers, $M_3 := \left\lceil \frac{1}{\mathrm{Cap}} \left(\sum_{k\in\mathcal{K}} w_k\right)\right\rceil$ is big enough. Constraints (52) force us to choose only arcs for the shortest path calculations that belong to established lines. Constraints (53) and (54) set the values of the node potential variables r_i^k and the shortest path indicators z_a^k. Note that if r_j^k and r_i^k denote shortest path distances from i to v_k, or from j to v_k respectively, $r_j^k + c_{ij} - r_i^k$ measures the difference between the shortest direct path from j to v_k and the shortest path from i to v_k that contains arc (i,j). The addition and subtraction of the constant M_4^{ij} avoids conflicts for arcs which are not in the routing network or which are not lying on shortest paths if M_4^{ij} is big enough, that is, if $M_4^{ij} \ge \left(\sum_{a\in A} c_a\right) + c_{ij}$. Constraints (55) set the distance from a destination node to itself to be 0 and hence initializes the implicit distance calculations. Constraints (56) act as linking constraint of the passenger flow assignment in (46-48) and the shortest path calculations in (52-56) allowing only arcs on shortest paths as candidates for the passenger flows.
This approach has first been described in [Sch14], where also a formal proof of correctness of this formulation has been given.

5 Solution Approaches

5.1 Integer Programming Approaches

To solve a bicriteria problem, i.e., to find the set of Pareto solutions with respect to travel time and costs, several scalarization approaches have been proposed in the literature (see, e.g., [Ehr06]). In the following, we focus on the ε-constraint method, where a budget B on the line costs is enforced.

Using different values for B, we can then construct a set of Pareto solutions. The ε-constraint method has been widely applied for line planning problems (see [Sch12b]).
In both formulations LPRC-1 and LPRC-2, we use sets of variables and constraints for every OD-pair, to ensure that this OD pair travels on a user-optimal shortest path. Clearly, these variables and constraints are not required if the user-optimal path is also a system-optimal path, i.e., when a solution to LPRA can be extended to a feasible solution to LPRC. We therefore consider two solution approaches, where not all variables and constraints for LPRC are included from the beginning.
In our first approach, we use lazy constraints to generate constraints (34)–(37) for LPRC-1, and constraints (52)–(56) for LPRC-2 on the fly during the solution process. That is, we start the solution process with model LPRA. Whenever Cplex finds a feasible solution, we check if any of these constraints are violated, i.e., if there is an OD-pair that does not travel on shortest paths. Finding such violated constraints can be done in polynomial time by shortest path computations. If it is the case that constraints are violated, the solution is rejected as being infeasible, and the set of constraints corresponding to the OD-pairs that do not travel on shortest paths are added to the current model.
In our second approach, we also begin with model LPRA, but do not add constraints on the fly during the solution process. Instead, we solve LPRA to optimality, and then check if this solution can be extended to a feasible solution of LPRC, i.e., if all passengers travel on shortest paths. If this is not the case, we add the sets of constraint corresponding to OD pairs that travel on sub-optimal paths, and solve the extended model again to optimality. This is repeated until an optimal solution to LPRC is found. In other words, we solve each model to optimality, before we check if every passenger travels on shortest paths and add new constraints, while the previous approach does so for every candidate solution in the solution process. Note that this approach does not produce any feasible solution to LPRC, until a solution is found that is both feasible and optimal.
The advantage of the second approach over the first approach is that less shortest path constraints are expected to be added to the model, as we only check solutions that are optimal in every iteration, and not any solution that is perceived as feasible. Hence, problem formulations can be kept smaller. The disadvantage is that several (mixed-)integer programs have to be solved, instead of a single problem as in the first approach.
To implement the first method in Cplex, we use callbacks to control the solution procedure. Internally, Cplex runs a branch-and-cut algorithm, and allows communication from the outside via such callbacks. In the incumbent callback (which is called whenever Cplex finds a new feasible solution), we run our shortest path algorithms to verify if every passenger travels on shortest paths. If this is not the case, the solution is rejected, and the corresponding OD-pairs are noted. In a lazy constraint callback (which is called by Cplex regularly), we then add all constraints corresponding to OD-pairs that have been noted in previous incumbent callbacks.

5.2 A Genetic Algorithm

As an alternative to these integer programming approaches, which aim at finding solutions with a certificate of optimality, we describe a metaheuristic algorithm that focusses on producing high-quality solutions without quality guarantees (which can be especially useful for larger-scale problems). There exist several genetic algorithms for biobjective problems, see, e.g., the overview [KCS06]. We follow the very successful NSGA-II algorithm type, see [DPAM02].
We reduce the solution space by making the following assumption: All passengers of one OD-pair use the same path. This is not required in the LPRC model; we therefore consider a smaller set of potential solutions, which may mean that some Pareto solutions cannot be found by our heuristic. However, this assumption has strong advantages as it simplifies the way we represent solutions.
A solution is encoded using only a binary vector s that denotes for each line $l \in \mathcal{L}$, if it is used or not. Such a vector is evaluated using the following algorithm. We first determine if s is feasible, by checking if this choice of lines allows every passenger to travel from origin to destination in the PTN. Using the PTN instead of the CGN reduces the computation time of such a feasibility

check. If s is feasible, we determine a shortest path for every OD-pair in the CGN, and then set line frequencies such that the resulting requirements on arc capacities are fulfilled.
The genetic algorithm begins with generating a random, feasible starting population of such vectors s. Additionally, we include the solution where all lines are build, i.e., $s_l = 1$ for all $l \in \mathcal{L}$. This solution gives the smallest possible travel time of all OD-pairs. Each solution is evaluated.
We then repeat the following procedures, until a time limit is reached: We find the non-dominated sorting, calculate the crowding distance, and crossover and mutate. Each step is explained in the following.
For non-dominated sorting, we first determine the subset of solutions which are not dominated by any other solution. These solutions are given rank 0, and are then removed for the purpose of this procedure. We find all solutions which are non-dominated for this reduced set of solutions, assign them rank 1, and so on, until each solution is given some rank. The rank is used to determine the fitness of a solution.
The crowding distance is used to ensure diversity by distributing solutions evenly over the Pareto frontier. Each set of solutions of equal rank can be ordered with respect to one of the two objective functions, yielding one or two neighbors for every solution. The crowding distance is the sum of distances in both objectives to these neighbors, normalized by the largest distance of all current solutions. This means that solutions for which other solutions of similar quality exist have a small crowding distance.
The crossover and mutation step uses rank and crowding distance in the following way. We choose two pairs of solutions at random from the current population. In each pair, we choose the solution which has the smaller rank value; if both have the same rank, we prefer the solution with the larger crowding distance. Having chosen one solution from each pair, these two solutions then generate an offspring by randomly combining their decisions to build lines or not. We then mutate the offspring by changing each decision with a small probability. If the resulting solution should be infeasible, the process is repeated. Once a feasible solution is found, it is evaluated. We generate offsprings this way using random pairs of solutions until the population size has doubled. In the next non-dominated sorting process, we only keep the half of the population which has the best rank.
As noted above, we include a solution that has the best possible travel time right from the beginning ($s_l = 1$ for all $l \in \mathcal{L}$), but this is not possible for the second objective, the line concept costs. We therefore help the genetic algorithm finding low-cost solutions with an additional local search, which is used with small probability on a new feasible offspring solution. The local search considers the $|\mathcal{L}|^2$ many moves that consist of flipping any pair of line decisions. If the resulting solution is feasible and cheaper than the previous solution, it is used instead, and the search process is repeated until no more improvements can be found.

6 Experiments

All experiments described in this section were conducted on a computer with a 16-core Intel Xeon E5-2670 processor, running at 2.60 GHz with 20MB cache, and Ubuntu 12.04. Processes were pinned to one core. We used Cplex v.12.6 [IBM13] to solve (mixed-)integer programs, and LEMON 1.2.3 [COI12] for shortest path computations.

6.1 Instances

We test our models on two sets of instances: On instances we randomly generated along realistic parameters, and on an instance modelling the German long distance rail network. Both sets serve different purposes. The randomly generated instances have the advantage that we can directly control their size, and we can generate a quantity that is large enough to allow significant comparisons between solution methods. The German network instance on the other hand is considerably larger and more representative for real-world applications of our model.

6.1.1 Random instances

We first describe the randomly generated instances, which mimic real-world problems. Let a number n of stations be given. Using a uniform distribution, we randomly assign each station $i \in \{1, \dots, n\}$ a set of coordinates (x_i, y_i) in the square $[0, 1]$. Distances between stations are then always given as the Euclidean distance in these coordinates. We then assign to each station a population value p_i by using the population of one of the 30 largest cities in Germany, chosen uniformly at random.
The total number of passengers is set to $1000 \cdot n$. The origin station of each passenger is chosen using a probability that is proportional to the population value p_i of each station. The destination of each passenger is then assigned using a gravity model, i.e., the probability that a station is chosen as destination is proportional to its population, and inverse proportional to its distance to the origin station. Note that $\mathcal{OD}$ matrices generated by this procedure are dense. This way, we can expect a realistic distribution of travel demand among stations.
We then complete the PTN by generating edges between stations. This is done by computing a Delaunay triangulation to ensure an even network design, similar to as it is used in practice. We then randomly remove arcs with a low probability to perturb the network. If the resulting graph should not be connected, this process is repeated.
We then generate a line pool, consisting of $3n$ lines. Each line is a random walk through the PTN, which chooses in each step a neighbor that was not visited before. After each step, the line stops with probability 70%. If a line generated this way does not consists of at least three stations, the process is repeated. The cost of a line the total distance it covers, multiplied with a random value from $[0.5, 1.5]$. The speedfactor of a line is a random value in $[1, 5]$. Every train has a capacity of 100.
For each value of n in $\{4, \dots, 10\}$, we generate 20 instances this way. We denote by $\mathcal{I}_n$ the set of instances with n stations. In Table 1, we summarize some graph properties of the resulting instances. Columns $|S|$ and $|E|$ give the average number of stations and directed arcs in the PTN, respectively. Column 'dens.' gives the average graph density, i.e., the actual number of edges divided by the number of potential edges in the graph. In 'deg.', we give the average node degree.

	PTN									CGN	
Set	$\|S\|$	$\|E\|$	dens.	deg.	betw.	diam.	l.p.p.	lines	l.len.	$\|V\|$	$\|A\|$
$\mathcal{I}_4$	4	9.7	0.81	4.9	0.23	1.9	4.19	12	3.0	52.0	132.0
$\mathcal{I}_5$	5	14.1	0.71	5.6	0.33	2.2	4.06	15	3.2	68.3	174.8
$\mathcal{I}_6$	6	18.6	0.62	6.2	0.34	2.2	4.19	18	3.3	84.1	216.2
$\mathcal{I}_7$	7	22.9	0.54	6.5	0.32	2.8	4.29	21	3.4	99.7	257.0
$\mathcal{I}_8$	8	28.3	0.51	7.1	0.33	2.9	4.26	24	3.4	112.8	290.3
$\mathcal{I}_9$	9	33.2	0.46	7.4	0.30	3.0	4.33	27	3.4	127.7	329.1
$\mathcal{I}_{10}$	10	38.1	0.42	7.6	0.30	3.3	4.40	30	3.4	141.5	364.5

Table 1: Average PTN and average CGN size for random instances.

Column 'betw.' shows the average betweenness of nodes, i.e., for every node, we count how often this node lies on a shortest path between any two other nodes. The values are normalized in $[0, 1]$. In 'diam.', the graph diameter is given, that is, the largest distance between any two nodes, where the distance is given by the length of the shortest path with unit arc lengths. In 'l.p.p.', we show the average number of lines per path. For every pair of nodes, we count how many different lines could potentially be used along a shortest path. In 'lines', the number of lines is given (by construction always equal to $3|S|$); and 'l.len.' gives the average number of edges in each line.
Finally, in $|V|$ and $|A|$, the number of nodes and edges in the resulting CGN is shown.

6.1.2 German long distance network

This problem instance models the long distance railway network of Germany. The PTN is taken from the LinTim toolbox [Lin16, GSS13a] and consists of 250 stations, 652 (directed) edges, and

132 lines. The edge density is 0.01, and average node degree is 5.2. The normalized average betweenness is 0.16, and the diameter is 27. The average number of lines per path is 53.1, and the average line length is 15.4. The corresponding CGN consists of 4,586 nodes and 12,744 arcs. The OD matrix has 48,842 non-zero entries.

6.2 Solution algorithms

Models LPRC-1 and LPRC-2 can be solved directly using a generic mixed-integer programming solver (in our experiments, we use Cplex [IBM13]). However, it may be that some passengers already travel on shortest paths when only the simpler problem LPRA was solved. We therefore consider the two solution approaches presented in Section 5.1 that do not include all constraints ensuring the usage of shortest paths from the beginning. These approaches can be used for both models, LPRC-1 and LPRC-2.
We use the following notation for these solution approaches:

- Solving LPRA directly with Cplex is denoted as CPX-A.
- Solving LPRC-1 directly with Cplex is denoted as CPX-1.
- Solving LPRC-1 using the first approach, which constructs violated lazy constraints on the fly in a single branch-and-bound tree, is denoted as CPX-1-L.
- Solving LPRC-1 using the second approach, which solves increasingly large problems iteratively to optimality, is denoted as CPX-1-It.
- Analogously, we write CPX-2, CPX-2-L, and CPX-2-It for LPRC-2.

6.3 Setup

In the following, we describe the outcomes of four experiments. The first three experiments use the smaller, randomly generated instances, while the last experiment uses the large real-world instance. We ask the following questions:

1. Which of the exact solution approaches for LPRC requires the least computation time?
2. How do user-optimal and system-optimal solutions compare, i.e., what are the differences between the solutions generated by models LPRA and LPRC?
3. How does the genetic algorithm scale with problem size, compared to the exact approaches?
4. Is it possible to find user-optimal solutions for the real-world instance?

Each question is answered in a separate experiment.

6.4 Experiment 1: Computation times for LPRC

We use the seven solution methods CPX-A, CPX-1/2, CPX-1/2-L and CPX-1/2-It to solve the 140 randomly generated instances $\mathcal{I}_4$ to $\mathcal{I}_{10}$. Each method is given a time limit of one hour. We measure the computation time to reach optimality, and how often optimality can be proven within the time limit.
To facilitate the comparison of these methods, we use a single budget B per instance, i.e., we calculate only one solution instead of the set of all Pareto efficient solutions. Generating a suitable budget B on the costs is therefore crucial for the following experiments. On the one hand, if B is too large, instances become easy to solve, and there may be no difference between solutions of models LPRA and LPRC, as there is enough capacity available to send every passenger on the shortest path the CGN allows. If B is too small, on the other hand, only few feasible solutions exist, and computation times become very large. To find a compromise between these two extremes, we first use the genetic algorithm (which does not require a budget B) for each instance. Let B^l be

the smallest budget of any Pareto solution found this way, and let B^u be the highest such budget. We set $B = \frac{1}{2}(B^l + B^u)$.
Our results are summarized in Tables 2 and 3.

Instance	CPX-A	CPX-1	CPX-1-It	CPX-1-L	CPX-2	CPX-2-It	CPX-2-L
$\mathcal{I}_4$	0.1	4.9	8.8	1.8	76.6	297.7	1148.4
$\mathcal{I}_5$	0.3	39.4	241.8	12.3	2585.0	2495.8	3417.6
$\mathcal{I}_6$	0.9	320.8	730.9	125.7	3448.0	3422.7	3600.0
$\mathcal{I}_7$	1.7	1112.5	1473.3	584.1	3600.0	3384.2	3600.0
$\mathcal{I}_8$	4.3	2687.6	3057.1	2135.9	3435.1	3392.3	3420.3
$\mathcal{I}_9$	8.9	3570.7	3600.0	3600.0	3600.0	3600.0	3600.0
$\mathcal{I}_{10}$	16.9	3600.0	3313.5	3457.9	3600.0	3600.0	3600.0

Table 2: Average computation times in seconds.

In Table 2, we show the average computation times in seconds over the 20 instances of each size. Note that 3600 seconds correspond to the time limit. Computation times are accompanied by the number of instances which could be solved to proven optimality within the time limit in Table 3. Clearly, CPX-A is several orders of magnitude faster than any other method, and could solve all instances within seconds. Comparing CPX-2 with the alternatives CPX-2-It and CPX-2-L, we find that using not all OD-pairs from the beginning does not lead to significant improvements – in fact, it is even unhelpful for the smallest instances.
This is different for CPX-1, where CPX-1-L tends to outperform CPX-1 and CPX-1-It. Overall, CPX-1-L shows best performance in solving model LPRC; furthermore, approaches based on CPX-1 outperform approaches based on CPX-2.

Instance	CPX-A	CPX-1	CPX-1-It	CPX-1-L	CPX-2	CPX-2-It	CPX-2-L
$\mathcal{I}_4$	20	20	20	20	20	19	15
$\mathcal{I}_5$	20	20	19	20	8	9	2
$\mathcal{I}_6$	20	20	19	20	1	1	0
$\mathcal{I}_7$	20	20	18	19	0	2	0
$\mathcal{I}_8$	20	8	4	13	1	2	1
$\mathcal{I}_9$	20	1	0	0	0	0	0
$\mathcal{I}_{10}$	20	0	2	1	0	0	0

Table 3: Number of optimal solutions.

6.5 Experiment 2: Comparison of solutions

For this experiment, we consider only optimal solutions from Experiment 1, independent of the solution approach that generated them. For instances where optimal solutions of both LPRA and LPRC are available, we measure the passenger travel time (i.e., the objective values of the models). However, these travel times reflect only part of the quality of a solution: Passengers in model LPRA may travel on paths that are not user-optimal, to enable a system-optimal solution. We use two additional quality measures to gauge the improvement in service quality a solution to LPRC delivers.
The first is the required additional capacity of arcs to enable shortest paths for all passengers. One may consider this value as the amount of missing capacity an operator would need to provide such that all passengers can travel on shortest paths, or, alternatively, as the amount of congestion of trains if all passengers do decide to travel on their shortest paths. This value is always 0 for solutions to LPRC.
The other quality measure is the amount of investment a line operator would need to make to improve the capacity of the line concept to a level that allows every passengers to use a system-

optimal path. For solutions to LPRC, this is simply the cost of the line concept. For LPRA, this is the cost of the line concept, plus possible additional costs to extend its capacity.
These two values are computed using mixed-integer programs. The congestion value is found by minimizing the following problem:

$$\min \sum_{a \in A} \gamma_a \tag{60}$$

$$s.t. \quad \sum_{a \in \delta^-(u_k^{\text{org}})} x_a^k = w_k \qquad \forall\ k \in \mathcal{K}, \tag{61}$$

$$\sum_{a \in \delta^+(v)} x_a^k - \sum_{a \in \delta^-(v)} x_a^k = 0 \qquad \forall\ k \in \mathcal{K},\ v \in V \setminus \{u_k, v_k\}, \tag{62}$$

$$\sum_{a \in \delta^+(v_k)} x_a^k = w_k \qquad \forall\ k \in \mathcal{K}, \tag{63}$$

$$\sum_{k \in \mathcal{K}} x_a^k \leq f_{l(a)} \text{Cap} + \gamma_a \qquad \forall\ a \in A, \tag{64}$$

$$\sum_{a \in A} c_a x_a^k = opt(F, k) \qquad \forall\ k \in \mathcal{K} \tag{65}$$

$$x_a^k \in \mathbb{N} \qquad \forall\ k \in \mathcal{K},\ a \in A, \tag{66}$$

$$f_l \in \mathbb{N} \qquad \forall\ l \in \mathcal{L} \tag{67}$$

$$\gamma_a \geq 0 \qquad \forall\ a \in A, \tag{68}$$

where $opt(F, k)$ denotes the length of a shortest route for OD-pair k in the network $\mathcal{N}(F)$, which is a precomputed constant in this formulation. For the cost of the line concept, we use a similar model that includes the costs of the lines.
To restrict total computation times, we use a time limit of one hour for the solution of these models via Cplex.
Table 4 shows the average travel time. In column LPRA, the total travel time of all passengers in the solution produced by LPRA is given, averaged over the instances of each size where optimal solutions for both LPRA and LPRC were found. The same averaged travel time for LPRC is shown in the next column. Column 'Relative' shows the relative increase in travel time from LPRA to LPRC in percent, while the last column 'Nr Inst' gives the number of instances over which the average is taken.
We find that travel time differences are especially large for the smallest instances. As there are only very few stations, they are densely connected, and the aspect of route assignment is particularly powerful. However, this evens out when the number of stations increases. For eight stations, the average increase in travel time is 8.5%, and for larger instances, there are not sufficiently many optimal solutions available to make a valid comparison.

Instance	LPRA	LPRC	Relative	Nr Inst
$\mathcal{I}_4$	480,174.8	606,540.4	30.2	20
$\mathcal{I}_5$	457,008.7	564,101.7	22.4	20
$\mathcal{I}_6$	573,905.4	706,699.1	25.6	20
$\mathcal{I}_7$	670,581.2	745,247.7	11.1	20
$\mathcal{I}_8$	741,944.6	804,882.6	8.5	14
$\mathcal{I}_9$	668,577.0	747,839.0	11.9	1
$\mathcal{I}_{10}$	1,167,376.5	1,201,710.5	3.0	2

Table 4: Average travel times.

Table 5 presents the service quality related measures. In the column 'Congestion', we give the objective value of model (60) – (68), i.e., the additional capacity required on arcs to enable user-optimal shortest paths for every OD-pair (recall that the capacity of every train is set to 100). The

next column shows how often a congestion value that was strictly larger than zero was observed. If the congestion is zero, we know that the solution of LPRA is also optimal to LPRC. This happened only once.

Instance	Congestion	Nr Cong.	LPRA-Budget	LPRC-Budget	Relative	Nr Inst
$\mathcal{I}_4$	1651.4	19	58.5	41.8	36.7	20
$\mathcal{I}_5$	1934.9	20	67.7	51.3	31.4	20
$\mathcal{I}_6$	3001.1	20	84.8	63.4	35.0	20
$\mathcal{I}_7$	3223.9	20	100.6	76.5	31.1	20
$\mathcal{I}_8$	2959.9	13	107.5	84.9	25.9	14
$\mathcal{I}_9$	365.1	1	140.7	98.1	43.4	1
$\mathcal{I}_{10}$	810.8	2	234.0	157.2	47.8	2

Table 5: Average congestion and average budget.

The following three columns, 'LPRA-Budget', 'LPRC-Budget' and 'Relative' present the average cost to build a line concept that enables user-optimal paths. LPRA-Budget gives the smallest budget when extending the LPRA solution using a similar formulation as in (60) – (68). LPRC-Budget is the cost of the LPRC solution, and 'Relative' gives the average relative increase from LPRC to LPRA in percent. As before, the last column gives the number of instances over which the average was taken.
The fact that in nearly all of our instances, users cannot travel on optimal paths without causing congested trains when using the LPRA solution underlines the importance of taking the route choice aspect into account. While the travel time ratio decreases with larger instances, this trend is not as clear in with regards to budgets. This means that savings are still relevant for larger instances, and speaks towards our model LPRC.

6.6 Experiment 3: Comparison of genetic algorithm with exact approach

In this experiment we compare the genetic algorithm with the best exact solution approach from Section 6.4, which is CPX-1-L. To this end, we run the genetic algorithm for ten minutes on every instance from $\mathcal{I}_4$ to $\mathcal{I}_{10}$. For every Pareto solution generated this way, we run CPX-1-L with another ten minutes time limit and the same line concept budget as the solution requires. We record if CPX-1-L is able to find a solution with a travel time that is at least as good as the travel time of the solution produced by the genetic algorithm. As the genetic algorithm may produce a large number of Pareto solutions, we restrict the comparison to 20 solutions per instance, which are chosen equidistantly over the Pareto front.
The genetic algorithm used a population of 2000 individuals for this experiment.
We summarize our findings in Table 6. In columns 'Better', 'Equal' and 'Worse', we give the average relative number of solutions in percent for which the exact method found a better, equal, or worse solution, respectively (i.e., for $\mathcal{I}_4$, CPX-1-L found a solution that dominates the corresponding solution produced by the genetic algorithm in 18.8% of all cases; found a solution of equal quality in 81.2% of cases; and never produced a solution of worse quality). The column 'Worse' also counts the cases where the exact method did not produce a feasible solution at all. Typically, solutions with larger budget were easier to reproduce.
The column 'Nr Sols' gives the average number of Pareto solutions produced by the genetic algorithm. There is a clear tendency that more Pareto solutions are produced for larger instances.
Table 6 indicates that the genetic algorithm is able to outperform our IP based approaches when the problem size increases. In particular, the genetic algorithm used the time limit of ten minutes to produce the complete Pareto front, while CPX-1-L was given ten minutes per solution to reproduce, which underlines the performance gap between both approaches.
To compare both approaches in more detail, we also show two columns $\epsilon_{IP \leq Gen}$ and $\epsilon_{Gen \leq IP}$. These values are calculated in the following way. Let $\mathcal{S}_{IP} \subset \mathbb{R}^2$ be the set of vectors of objective

Instance	Better	Equal	Worse	Nr Sols	$\epsilon_{IP \leq Gen}$	$\epsilon_{Gen \leq IP}$
$\mathcal{I}_4$	18.8	81.2	0.0	8.7	1.00	1.03
$\mathcal{I}_5$	23.7	74.2	2.1	13.3	1.01	1.03
$\mathcal{I}_6$	29.0	52.3	18.7	23.0	1.05	1.03
$\mathcal{I}_7$	20.6	32.1	47.4	26.8	1.16	1.02
$\mathcal{I}_8$	11.6	19.3	69.2	25.9	1.28	1.01
$\mathcal{I}_9$	6.7	7.4	86.0	47.8	1.51	1.01
$\mathcal{I}_{10}$	9.3	5.0	85.7	66.4	1.51	1.01

Table 6: Average percentage of solutions generated by genetic algorithm for which CPX-1-L finds a better, equal, or worse solution, and average number of solutions generated by genetic algorithm (Nr Sols).

values $(\sum_{k \in \mathcal{K}} \sum_{a \in \mathcal{A}} c_a x_a^k, \sum_{l \in \mathcal{L}} f_l b_l)$ found by using the IP method CPX-1-L for some instance I. Analogously, $\mathcal{S}_{Gen}$ is the set of objective values found by the genetic algorithm. To compute $\epsilon_{IP \leq Gen}$, we find the smallest $\epsilon \geq 0$ such that when all points of $\mathcal{S}_{Gen}$ are multiplied with ϵ, then every point of $\mathcal{S}_{Gen}$ is dominated by some point of $\mathcal{S}_{IP}$. More formally, we set

$$\epsilon_{IP \leq Gen}(I) := \min\{\epsilon : \forall (c_i, d_i) \in \mathcal{S}_{Gen}\ \exists (c_j, d_j) \in \mathcal{S}_{IP} \text{ s.t. } c_j \leq \epsilon c_i \text{ and } d_j \leq \epsilon d_i\}$$

The value $\epsilon_{IP \leq Gen}$ is then the average of $\epsilon_{IP \leq Gen}(I)$ over all instances of the respective set. Analogously for $\epsilon_{Gen \leq IP}$. A larger value of $\epsilon_{A \leq B}$ thus indicates worse performance of A. We see that while CPX-1-L performs slightly better than the genetic algorithm on small instances, it is outperformed already from $\mathcal{I}_6$ upwards.

6.7 Experiment 4: Real-world instance

As discussed in the previous experiment, our IP methods are not competitive for larger instances. This is especially the case for our considerably larger real-world instance, where both LPRA and LPRC are not able to produce any solution within reasonable time limit.

We run the genetic algorithm with a time limit of one hour, and a population size of 100. For this instance, we do not use the local search for improvements in the budget, as it may take a too large portion of computation time. Within one hour, our implementation was able to perform 25 iterations this way.

In Figure 3, we show the starting population in gray, and the non-dominated solutions after the last iteration in black. By construction, a best possible solution with respect to travel time is already included in the set of starting solutions (in the bottom right corner). The algorithm was able to significantly reduce the line concept costs, without a large increase in travel times. A sharp increase in travel times below a budget of approximately 64,000 can be observed: In this experiment, the solution with costs $64,033$ and travel time $1.42 \cdot 10^9$ is a strong candidate to consider for the line operator.

Overall, we find that our algorithm is able to solve LPRC on instances of real-world size within a reasonable time frame for the strategic planning stage. The solution set generated by the genetic algorithm still changed in the last iteration that was performed within the time limit, which means that small further improvements are possible.

7 Conclusions and further research

The economic success of a line operator depends on plenty of factors, and service quality is amongst the most prominent of them, as it has direct impact on customer experience. Previous models include short passenger travel times in a system-optimal way, i.e., they use the assumption that a path can be assigned to each passenger. In this paper we considered models which do not require this unrealistic assumption: Each passenger chooses a path which minimizes his or her travel time.

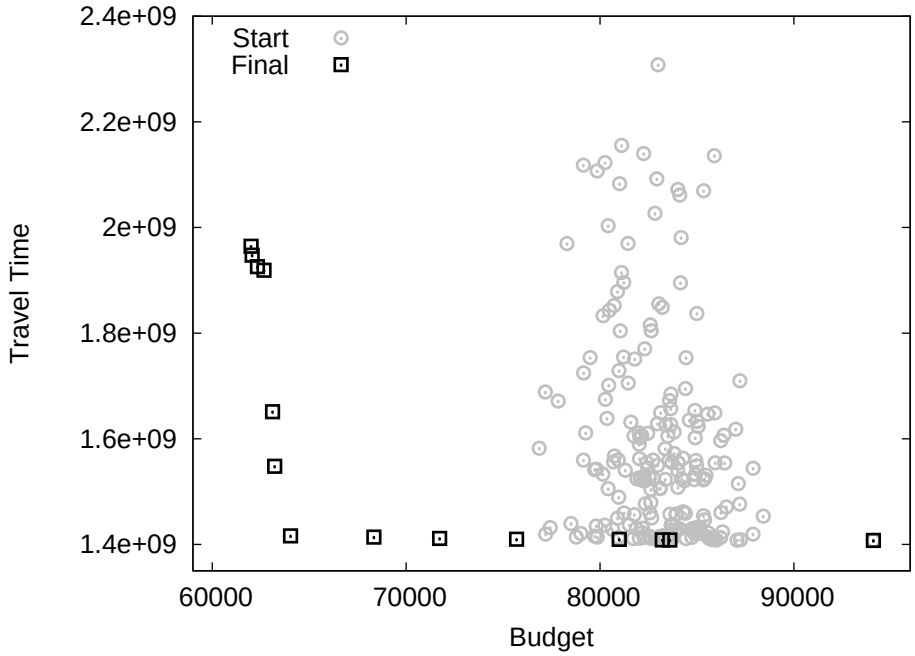


Figure 3: Solution output of genetic algorithm for real-world instance.

If several shortest paths exist, we still assume that we can assign one of them (e.g., by using online route information systems).
We formulate the resulting problem with two integer programs, which are considerably larger than the route assignment model. We therefore propose decomposition approaches that do not have to create the complete model at the beginning of the solution process, and find in computational experiments on randomly generated instances that such an approach is able to reduce computation times.
However, our integer programming approaches do not scale to large-scale problems with hundreds of stations. To this end, we propose a genetic solution algorithm. In our experiments we showed that this algorithm is competitive for smaller instances, and is able to produce solutions for problems with large size.
An alternative way to formulate the route choice model using less variables than LPRC-2 is shown in Appendix A. In our experiments, this approach showed inferior performance compared to the other models we presented. Further research is required to better understand why this is the case.
Finally, more complex transfer penalties p may be considered, that also depend on the frequencies of the lines involved. While our MIPs would become nonlinear if arc lengths c_a depended on f_l, it may be possible to include for the genetic algorithm with suitable extension.

References

[AFTÜ09] A. Altin, B. Fortz, M. Thorup, and H. Ümit. Intra-domain traffic engineering with shortest path routing protocols. *4OR*, 4:301–335, 2009.

[BCCP09] B. Beltran, S. Carrese, E. Cipriani, and M. Petrelli. Transit network design with allocation of green vehicles: A genetic algorithm approach. *Transportation Research Part C: Emerging Technologies*, 17(5):475–483, 2009.

[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 M. Hickman, P. Mirchandani, and S. Voß, editors, *Computer-aided Systems in Public Transport*, volume 600 of *Lecture Notes in Economics and Mathematical Systems*, pages 363–378. Springer Berlin Heidelberg, 2008.

[BK12] R. Borndörfer and M. Karbstein. A direct connection approach to integrated line planning and passenger routing. In *12th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS)*, volume 25 of *OpenAccess Series in Informatics (OASIcs)*, pages 47–57. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2012.

[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.

[BLMS01] L. Brotcorne, M. Labbé, P. Marcotte, and G. Savard. A bilevel model for toll optimization on a multicommodity transportation network. *Transportation Science*, 35(4):345–358, 2001.

[BS02] T. Basar and R. Srikant. A stackelberg network game with a large number of followers. *Journal of Optimization Theory and Applications*, 115:479–490, 2002.

[BS12] P. H. Bovy and E. Stern. *Route Choice: Wayfinding in Transport Networks: Wayfinding in Transport Networks*, volume 9. Springer Science & Business Media, 2012.

[Bus98] M. Bussieck. *Optimal Lines in Public Rail Transport*. PhD thesis, Technische Universität Braunschweig, 1998.

[CF95] I. Constantin and M. Florian. Optimizing frequencies in a transit network: a nonlinear bi-level programming approach. *International Transactions in Operational Research*, 2(2):149–164, 1995.

[CMS07] B. Colson, P. Marcotte, and G. Savard. An overview of bilevel optimization. *Annals of Operations Research*, 153(1):235–256, 2007.

[COI12] COIN-OR. LEMON Graph Library 1.2.3, 2012. See http://lemon.cs.elte.hu.

[CR11] R. Cordone and F. Redaelli. Optimizing the demand captured by a railway system with a regular timetable. *Transportation Research Part B: Methodological*, 45(2):430 – 446, 2011.

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

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

[dMI06] L. dell'Olio, J. Moura, and A. Ibeas. Bi-level mathematical programming model for locating bus stops and optimizing frequencies. *Transportation Research Record: Journal of the Transportation Research Board*, 1971:23–31, 2006.

[DPAM02] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *Evolutionary Computation, IEEE Transactions on*, 6(2):182–197, Apr 2002.

[DSSW09] D. Delling, P. Sanders, D. Schultes, and D. Wagner. Engineering route planning algorithms. In *Algorithmics of large and complex networks*, pages 117–139. Springer, 2009.

[Ehr06] M. Ehrgott. *Multicriteria optimization*. Springer Science & Business Media, 2006.

[FM06] W. Fan and R. Machemehl. Optimal transit route network design problem with variable transit demand: Genetic algorithm approach. *Journal of Transportation Engineering*, 132(1):40–51, 2006.

[FMSR13] 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.

[GSS04] Z. Gao, H. Sun, and L. L. Shan. A continuous equilibrium network design model and algorithm for transit systems. *Transportation Research Part B: Methodological*, 38(3):235–250, 2004.

[GSS13a] M. Goerigk, M. Schachtebeck, and A. Schöbel. Evaluating line concepts using travel times and robustness. *Public Transport*, 5(3):267–284, 2013.

[GSS+13b] M. Goerigk, M. Schmidt, A. Schöbel, M. Knoth, and M. Müller-Hannemann. The price of strict and light robustness in timetable information. *Transportation Science*, 48(2):225–242, 2013.

[GvHK04] J.-W. Goossens, S. van Hoesel, and L. Kroon. A branch-and-cut approach for solving railway line planning problems. *Transportation Science*, 38(3):379–393, 2004.

[GvHK06] J.-W. Goossens, S. van Hoesel, and L. Kroon. On solving multi-type railway line planning problems. *European Journal of Operational Research*, 168(2):403 – 424, 2006. Feature Cluster on Mathematical Finance and Risk Management.

[GYW06] 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.

[Har16] J. Harbering. *Planning a Public Transportation System with a View Towards Passengers' Convenience*. PhD thesis, University of Göttingen, 2016.

[HBK15] H. Hoppmann, R. Borndörfer, and M. Karbstein. Timetabling and passenger routing in public transport. In *Proceedings of Conference on Advanced Systems in Public Transport 2015 (CASPT2015)*, 2015. accepted for publication.

[IBM13] IBM. *IBM ILOG CPLEX 12.6 User's Manual*, 2013.

[KCS06] A. Konak, D. W. Coit, and A. E. Smith. Multi-objective optimization using genetic algorithms: A tutorial. *Reliability Engineering & System Safety*, 91(9):992–1007, 2006.

[KLO97] Y. A. Korilis, A. A. Lazer, and A. Orda. Achieving network optima using stackelberg routing strategies. *IEEE/ACM Transactions on Networking*, 5:161, 1997.

[KMN14] L. Kroon, G. Maróti, and L. Nielsen. Rescheduling of railway rolling stock with dynamic passenger flows. *Transportation Science*, 49(2):165–184, 2014.

[KV04] B. Y. Kara and V. Verter. Designing a road network for hazardous materials transportation. *Transportation Science*, 38(2):188–196, 2004.

[Lin16] LinTim - Integrated optimization in public transportation, 2016. [Online, http://lintim.math.uni-goettingen.de; accessed March 2016].

[LMMO07] G. Laporte, Á Marín, J.A. Mesa, and F.A. Ortega. An integrated methodology for the rapid transit network design problem. In *Algorithmic Methods for Railway Optimization*, volume 4359 of *Lecture Notes in Computer Science*, pages 187–199. Springer Berlin Heidelberg, 2007.

[LMOS05] G. Laporte, J. A. Mesa, F. A. Ortega, and I. Sevillano. Maximizing trip coverage in the location of a single rapid transit alignment. *Annals of Operations Research*, 136(1):49–63, 2005.

[LSC+14] H. Lee, Y. Song, S. Choo, K.-Y. Chung, and K.-D. Lee. Bi-level optimization programming for the shipper-carrier network problem. *Cluster Computing*, 17(3):805–816, 2014.

[MHSWZ07] M. Müller-Hannemann, F. Schulz, D. Wagner, and C. Zaroliagis. Timetable information: Models and algorithms. In *Algorithmic Methods for Railway Optimization*, pages 67–90. Springer, 2007.

[Mig95] A. Migdalas. Bilevel programming in traffic planning: models, methods and challenge. *Journal of Global Optimization*, 7(4):381–405, 1995.

[NJ08] K. Nachtigall and K. Jerosch. Simultaneous network line planning and traffic assignment. In M. Fischetti and P. Widmayer, editors, *8th Workshop on Algorithmic Approaches for Transportation Modeling, Optimization, and Systems*, 2008.

[OW11] J. de D. Otuzar and L.G. Willumsen. *Modelling Transport*. J. Wiley & Sons Inc., Chichester u. a. O., 4 edition, 2011.

[Pat15] M. Patriksson. *The traffic assignment problem: models and methods*. Courier Dover Publications, 2015.

[PB06] M. Pfetsch and R. Borndörfer. Routing in line planning for public transport. In H.-D. Haasis, H. Kopfer, and J. Schönberger, editors, *Operations Research Proceedings 2005*, volume 2005 of *Operations Research Proceedings*, pages 405–410. Springer Berlin Heidelberg, 2006.

[Rou04] T. Roughgarden. Stackelberg scheduling strategies. *SIAM journal on computing*, 33:332, 2004.

[Rou05] T. Roughgarden. *Selfish Routing and the Price of Anarchy*. Massachussetts Institute of Technology, 2005.

[Sch05] S. Scholl. *Customer-Oriented Line Planning*. PhD thesis, Technische Universität Kaiserslautern, 2005. published by dissertation.de.

[Sch12a] M. Schmidt. *Integrating Routing Decisions in Network Problems*. PhD thesis, Universität Göttingen, 2012.

[Sch12b] A. Schöbel. Line planning in public transportation: models and methods. *OR spectrum*, 34(3):491–510, 2012.

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

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

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

[SK12] H. Shimamoto and F. Kurauchi. Optimisation of a bus network configuration and frequency considering the common lines problem. *Journal of Transportation Technologies*, 2(03):220, 2012.

[SMFZ10] H. Shimamoto, N. Murayama, A. Fujiwara, and J. Zhang. Evaluation of an existing bus network using a transit network optimisation model: a case study of the hiroshima city bus network. *Transportation*, 37(5):801–823, 2010.

[SS06] A. Schöbel and S. Scholl. Line planning with minimal transfers. In *5th Workshop on Algorithmic Methods and Models for Optimization of Railways*, number 06901 in Dagstuhl Seminar Proceedings, 2006.

[SS12] M. Schmidt and A. Schöbel. The complexity of integrating routing decisions in public transportation models. Technical report, Institute for Numerical and Applied Mathematics, University of Goettingen, 2012.

[SS14] M. Schmidt and A. Schöbel. Location of speed-up subnetworks. *Annals of Operations Research*, 223(1):379–401, 2014.

[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.

[vKM16] E. van der Hurk, L. Kroon, and G. Maroti. Passenger guidance and rolling stock rescheduling under uncertainty for disruption management. submitted, 2016.

[WN13] N. H.M. Wilson and A. Nuzzolo. *Schedule-based dynamic transit modeling: theory and applications*, volume 28. Springer Science & Business Media, 2013.

[Woo11] R. K. Wood. Bilevel network interdiction models: Formulations and solutions. *Wiley Encyclopedia of Operations Research and Management Science*, 2011.

A Appendix: An alternative model for LPRC

In model LPRC-2, we use a set of flow variables x^k for each OD-pair $k \in \mathcal{K}$. Such flows can be used to complete a whole shortest path tree from every origin instead, meaning that the number of flow variable sets can be reduced from $\mathcal{O}(n^2)$ to $\mathcal{O}(n)$.
Let $\mathcal{O}$ the set of all origin nodes, i.e., $\mathcal{O} = \{u \in V : \exists v \in V \text{ with } (u,v) \in \mathcal{OD}\}$, and let $\mathcal{D}(u)$ be the set of all destinations for which there is demand from node $u \in \mathcal{O}$. We give each node in $\mathcal{O}$ an index $k = 1, \dots, \mathcal{K}' = |\mathcal{O}|$, and denote by $w(u,v)$ the OD-demand from node u to node v. The following integer program is then a more compact version of LPRC-2.

$$\min \sum_{k \in \mathcal{K}'} \sum_{a \in A} c_a x_a^k,$$

$$s.t. \quad \sum_{a \in \delta^+(v)} x_a^k - \sum_{a \in \delta^-(v)} x_a^k = \begin{cases} -\sum_{v' \in \mathcal{D}(u_k)} w(u_k, v') & \text{if } v = u_k \\ w(u_k, v) & \text{if } v \in \mathcal{D}(u_k) \\ 0 & \text{otherwise} \end{cases} \qquad \forall\ u_k \in \mathcal{O},\ v \in V$$

$$\sum_{k \in \mathcal{K}'} x_a^k \le f_{l(a)} \mathrm{Cap} \qquad \forall\ a \in A,$$

$$\sum_{l \in \mathcal{L}} f_l b_l \le B,$$

$$y_l \le f_l \qquad \forall\ l \in \mathcal{L},$$

$$M_3 \cdot y_l \ge f_l \qquad \forall\ l \in \mathcal{L},$$

$$z_a^k \le y_{l(a)} \qquad \forall\ a \in A,\ k \in \mathcal{K}',$$

$$r_j^k + c_{ij} - r_i^k + z_{ij}^k + M_4^{ij}(1 - y_{l(i,j)}) \ge 1 \qquad \forall\ (i,j) \in A,\ k \in \mathcal{K}',$$

$$r_j^k + c_{ij} - r_i^k - M_4^{ij} \cdot (1 - z_{ij}^k) \leq 0 \qquad \forall\ (i,j) \in A,\ k \in \mathcal{K}',$$

$$\sum_{v \in \mathcal{D}(u_k)} r_v^k = 0 \qquad \forall\ u_k \in \mathcal{O},$$

$$x_a^k \leq z_a^k \cdot \sum_{v \in \mathcal{D}(u_k)} w(u_k, v) \qquad \forall\ a \in A,\ u_k \in \mathcal{O},$$

$$r_i^k \in \mathbb{N} \qquad \forall\ k \in \mathcal{K}',\ i \in V,$$

$$x_a^k \in \mathbb{N}, z_a^k \in \{0,1\} \qquad \forall\ k \in \mathcal{K}',\ a \in A,$$

$$f_l \in \mathbb{N}, y_l \in \{0,1\} \qquad \forall\ l \in \mathcal{L}.$$

The same approach can also be used to formulate a model with a set of flow variables for every destination, instead of origin; furthermore, even though we presented the model using LPRC-2 as a starting point, the same ideas can also be applied to rewrite model LPRC-1.