Look-Ahead Approaches for Integrated Planning in Public Transportation*

Julius Pätzold[1], Alexander Schiewe[2], Philine Schiewe[3], and Anita Schöbel[4]

1 **Institut für Numerische und Angewandte Mathematik, Universität Göttingen, Göttingen, Germany**
`j.paetzold@math.uni-goettingen.de`

2 **Institut für Numerische und Angewandte Mathematik, Universität Göttingen, Göttingen, Germany**
`a.schiewe@math.uni-goettingen.de`

3 **Institut für Numerische und Angewandte Mathematik, Universität Göttingen, Göttingen, Germany**
`p.schiewe@math.uni-goettingen.de`

4 **Institut für Numerische und Angewandte Mathematik, Universität Göttingen, Göttingen, Germany**
`schoebel@math.uni-goettingen.de`

Abstract

In this paper we deal with three consecutive planning stages in public transportation: Line planning (including line pool generation), timetabling, and vehicle scheduling. These three steps are traditionally performed one after another in a sequential way often leading to high costs in the (last) vehicle scheduling stage. In this paper we propose three different ways to "look ahead", i.e., to include aspects of vehicle scheduling already earlier in the sequential process: an adapted line pool generation algorithm, a new cost structure for line planning, and a reordering of the sequential planning stages. We analyze these enhancements experimentally and show that they can be used to decrease the costs significantly.

1998 ACM Subject Classification G.1.6 Optimization, G.2.2 Graph Theory, G.2.3 Applications

Keywords and phrases line pool generation, line planning, vehicle scheduling, integrated planning, public transport

Digital Object Identifier 10.4230/OASIcs.ATMOS.2017.17

1 Sequential versus integrated planning

Planning a public transport supply can have many goals. Two major goals are usually minimizing the perceived travel times of passengers as well as the costs that incur to the public transportation company. Motivated by this we consider a bi-objective model for railway or bus planning with these two objectives.

Traditionally, public transportation planning is done in sequential stages. The first stage after the design of a network, that is spanned by stops (or stations) and their direct connections (edges or tracks), is *line planning*. In this stage, first a set of possible lines, the line pool, has to be generated on the network. Research towards the effect of line pool

* This work was partially supported by DFG under SCHO 1140/8-1 and by the Simulation Science Center Clausthal/Göttingen.

17th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2017).
Editors: Gianlorenzo D'Angelo and Twan Dollevoet; Article No. 17; pp. 17:1–17:16
Open Access Series in Informatics
OASICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

generation, and an algorithm to find suitable line pools is presented in [7]. In the line planning problem one then chooses a feasible subset of lines from the line pool, i.e., a set of lines such that all passengers can be transported. See [21] for an overview. With a given line plan one can create an event-activity network which constitutes the input for the *timetabling stage*. Periodic timetabling consists of deciding when and how fast vehicles (trains or buses) should drive along the edges and how long they should wait at stops (or stations). The problem is modeled as a periodic event scheduling problem (PESP), see [23]. Other timetabling models can be found in [10]. After a timetable is chosen, vehicle schedules are planned, determining which vehicle should drive which route such that all lines are operated according to their timetables. A survey on *vehicle scheduling* is given in [4]. Finally, crew scheduling and rostering are planning stages to be performed after the vehicle schedules are found.

Obviously, proceeding sequentially does not need to lead to an optimal solution as there are dependencies between the different subproblems. It would hence be beneficial to solve the entire problem in an integrated system. Since this is computationally too complex, heuristic approaches have been proposed as in [22].

Our contribution. We consider line planning, timetabling and vehicle scheduling in conjunction with each other. To this end we formally define what an *integrated* transport supply (LTS-plan), consisting of a line plan, a timetable, and a vehicle schedule, is and how it can be evaluated. We propose three enhancements of the traditional approach which consider the vehicle scheduling costs already in the line planning stage. Finally, we evaluate them experimentally and show that our proposed enhancements lead to LTS-plans with significantly smaller costs than the traditional sequential approach.

2 A bi-objective model for integrated planning in public transportation

In this section we formally describe what a feasible transport supply (*LTS-plan*), consisting of a line plan (L), a timetable (T), and a vehicle schedule (S), is and how its quality can be evaluated. Note that for the single stages, i.e., for a line plan, for a timetable, and for a vehicle schedule, this has been extensively discussed in the literature. However, it is in the literature usually assumed that an event-activity network is already known for timetabling and a set of trips is already given for vehicle scheduling. Since we plan from scratch, we also have to describe the intermediate steps, i.e., how to build the event-activity network and how to build the set of trips. In order to keep the timetabling step tractable, we restrict ourselves in this paper to periodic LTS-plans for which all lines are operated with the same frequency.

As input for the bi-objective model we are given:

- A public transport network PTN$=(V,E)$ consisting of a set of stops V and direct connections E between them.
- For every node $v \in V$:
  - lower and upper bounds $L_v^{wait} \leq U_v^{wait}$ for the time vehicles wait at stop v,
  - lower and upper bounds $L_v^{trans} \leq U_v^{trans}$ for the time passengers need to transfer between two vehicles at the same stop v.
  - We furthermore need for every pair $v, u \in V$ the time(v,u) a vehicle needs if it drives directly from stop v to stop u.
- For every edge $e = (v_1, v_2) \in E$:
  - a length (in kilometers) length_e,
  - lower and upper edge frequency bounds $f_e^{\min} \leq f_e^{\max}$,
  - lower and upper bounds on the travel times along the edge, i.e., $L_e^{drive} \leq U_e^{drive}$.

- An OD-matrix W with entries W_{uv} for each pair of stops $u, v \in V$. The OD-matrix is assumed to be consistent with the lower edge frequencies, i.e., there exist paths P_{uv} for every OD-pair (u, v) through the PTN such that for every edge e we have:

$$\sum_{u,v \in V:\, e \in P_{uv}} W_{uv} \leq \mathrm{Cap} \cdot f_e^{\min}$$

  for Cap being the capacity of the (identical) vehicles, i.e., each passenger can be transported,
- a period length T, and the number of periods p to be considered for planning
- a penalty pen for transfers,
- a minimal turnaround time for vehicles $L_{\min}$,
- cost parameters
  - c_1 costs per minute for a vehicle driving with passengers,
  - c_2 costs per kilometer for a vehicle driving with passengers,
  - c_3 costs per vehicle for the whole planning horizon (p periods),
  - c_4 costs per minute for a vehicle driving empty (i.e., without passengers),
  - c_5 costs per kilometer for a vehicle driving empty (i.e., without passengers).

We then look for an LTS-plan, which consists of a *line plan* (L), a periodic *timetable* (T) and a *vehicle schedule* (S) which are together feasible. These objects are defined as follows:

Line plan L

A *line* is a path through the PTN. A *line plan* is a set of lines $\mathcal{L}$, which is feasible if

$$f_e^{\min} \leq |\{l \in \mathcal{L} : e \in l\}| \leq f_e^{\max}, \tag{1}$$

i.e., if each edge of the PTN is covered by the required number of lines. We assume that lines are symmetric, i.e., they are operated in both directions. In our setting all lines are operated with a frequency of 1.

Timetable T

Given a set of lines, a timetable assigns a time to every departure and arrival of every line at its stops. These times are then repeated periodically. In order to model a timetable usually event-activity networks $\mathcal{N} = (\mathcal{E}, \mathcal{A})$ are used (see, e.g., [11, 12, 14, 17, 18]). The set of events $\mathcal{E}$ consists of all departures and all arrivals of all lines at all stops, and the set $\mathcal{A}$ connects these events by driving, waiting and transfer activities. For each activity, the number of passengers using this activity is usually given as input for timetabling. (It is subject of ongoing research how this can be relaxed, see [3, 6, 19, 20]). The lower and upper bounds L_a and U_a are set as

- L_e^{drive} and U_e^{drive} if a is a driving activity on edge $e \in E$,
- L_v^{wait} and U_v^{wait} if a is a waiting activity in stop $v \in V$, and as
- L_v^{trans} and U_v^{trans} if a is a transfer activity in stop $v \in V$.

A timetable π is an assignment of times $\pi_j \in \mathbb{Z}$ to every event $j \in \mathcal{E}$. It is feasible if it respects the lower and upper bounds for all its activities, i.e., if

$$(\pi_j - \pi_i - L_a) \mod T \in [0, U_a - L_a] \text{ for all } a = (i, j) \in \mathcal{A}. \tag{2}$$

The objective function in timetabling minimizes the total slack times. If all passengers use the paths they have been assigned to in the event-activity network this is equivalent to minimizing the sum of passengers' travel times.

Vehicle schedule S

Given a set of lines and a timetable, a *vehicle schedule* determines the number of vehicles and the exact routes of the vehicles for operating the timetable. To this end, we use the line plan and the timetable to construct a set of *trips* $\mathcal{T}$ where each trip

$$t = (l_t, v_t^{start}, v_t^{end}, \tilde{\pi}_t^{start}, \tilde{\pi}_t^{end}) \in \mathcal{T}$$

is specified by a line l_t together with its first and last stop v_t^{start} and v_t^{end} and its corresponding *start time* $\tilde{\pi}_t^{start}$ and *end time* $\tilde{\pi}_t^{end}$. These times can be taken from the periodic timetable, but we have to consider the real time (e.g. in minutes after midnight) by adding the correct multiple of the period length. The end time $\tilde{\pi}_t^{end}$ of a line at its final stop is the arrival time at this stop plus some minutes allowing passengers to deboard. Analogously, the start time $\tilde{\pi}_t^{start}$ of a line at a stop is the time when it arrives at this stop, i.e., a bit earlier than its departure time there. For every line l we receive two trips starting per period, namely one forward and one backward trip. A route of a vehicle is given by its sequence of trips $r = (t_1, \ldots, t_k)$ such that

$$(\tilde{\pi}_{t_{i+1}}^{start} - \tilde{\pi}_{t_i}^{end}) \geq \text{time}(v_{t_i}^{end}, v_{t_{i+1}}^{start}) \text{ for all } i = 1, \ldots, k-1.$$

A set of vehicle routes $\mathcal{R}$ is feasible if all its routes are feasible and if each trip is contained in exactly one route.

Evaluating an LTS-plan

An LTS-plan is specified by a line plan, a corresponding timetable and a corresponding vehicle schedule, i.e., it is specified by the tuple $(\mathcal{L}, \pi, \mathcal{R})$. Given a feasible LTS-plan we use the two most common evaluation criteria: the sum of passengers' travel times (including a penalty for every transfer) and the costs. These objectives are formally defined below:

Costs. The costs of an LTS-plan depend mainly on the costs of the corresponding vehicle schedule and thus on the distance which is driven, the total duration of driving and the number of required vehicles. For the distance and the duration of the trips we distinguish if the vehicle drives on a trip which can be used by passengers (here called *full ride*) or if the vehicle drives empty between two consecutive trips t_i, t_{i+1} in the same vehicle route (here called an *empty ride*) as the costs can be different for full and empty rides.

As the vehicle schedule in general is aperiodic, we consider the costs for a whole planning horizon (e.g. a day) instead of a planning period by rolling out the periodic line plan and timetable for a fixed time span which is given by the number of periods p it covers. Note that we have to take special care at the beginning and the end of the roll-out period, regarding lines traversing the period boundaries. For simplicity reasons we do not go into detail here how this is handled explicitly.

Before defining the costs, we introduce the duration and the length of a line and an empty ride. Let a line be defined as a sequence of nodes and edges.

The duration of a line can be determined after the timetable is known. We get

$$\begin{aligned}
\text{dur}_l = &\sum_{\substack{a=(i,j)\in\mathcal{A}_{drive}:\\ a \text{ belongs to } e\in l}} (L_e^{drive} + (\pi_j - \pi_i - L_e^{drive} \mod T)) \\
&+ \sum_{\substack{a=(i,j)\in\mathcal{A}_{wait}:\\ a \text{ belongs to } v\in l}} (L_v^{wait} + (\pi_j - \pi_i - L_v^{wait} \mod T)),
\end{aligned}$$

i.e., all driving times along edges and waiting times at stops are added. When a heuristic approach to timetabling is used where the duration of all driving and waiting activities is set to their respective lower bounds, as done here, the duration of a line simplifies to

$$\mathrm{dur}_l = \sum_{e \in l} L_e^{drive} + \sum_{v \in l} L_v^{wait}. \tag{3}$$

The length of a line is computed as sum over all edge lengths

$$\mathrm{length}_l = \sum_{e \in l} \mathrm{length}_e$$

and is independent from the timetable. The duration of an empty ride between two trips $t_1 = (l_{t_1}, v_{t_1}^{start}, v_{t_1}^{end}, \tilde{\pi}_{t_1}^{start}, \tilde{\pi}_{t_1}^{end})$ and $t_2 = (l_{t_2}, v_{t_2}^{start}, v_{t_2}^{end}, \tilde{\pi}_{t_2}^{start}, \tilde{\pi}_{t_2}^{end})$ can be computed as

$$\mathrm{dur}_{t_1,t_2} = \tilde{\pi}_{t_2}^{start} - \tilde{\pi}_{t_1}^{end},$$

i.e., the time between the end of t_1 and the start of t_2.

The length of the empty ride is defined as

$$\mathrm{length}_{t_1,t_2} = SP(v_{t_1}^{end}, v_{t_2}^{start}),$$

i.e., we assume that a vehicle takes the shortest path from the last station $v_{t_1}^{end}$ of trip t_1 to the first station $v_{t_2}^{start}$ of trip t_2.

Now we can define the following cost components. Note that we have to count the full duration and length of each line twice as two trips belong to every line (one in forward and one in backward direction).

- *full duration*, i.e., time it takes to cover all trips (full rides):

$$\mathrm{dur}_{\mathrm{full}} = \sum_{l \in \mathcal{L}} 2 \cdot \mathrm{dur}_l \cdot \mathrm{p}\,,$$

- *full distance*, i.e., distance driven along lines:

$$\mathrm{length}_{\mathrm{full}} = \sum_{l \in \mathcal{L}} 2 \cdot \mathrm{length}_l \cdot \mathrm{p}\,,$$

- *number of vehicles*: $\mathrm{veh} = |\mathcal{R}|\,,$
- *empty duration*, i.e., time of empty rides between trips:

$$\mathrm{dur}_{\mathrm{empty}} = \sum_{r=(t_1,\dots,t_{k_r}) \in \mathcal{R}} \sum_{i=1}^{k_r-1} \mathrm{dur}_{t_i,t_{i+1}}\,,$$

- empty distance, i.e., distance of empty rides between trips:

$$\mathrm{length}_{\mathrm{empty}} = \sum_{r=(t_1,\dots,t_{k_r}) \in \mathcal{R}} \sum_{i=1}^{k_r-1} \mathrm{length}_{t_i,t_{i+1}}\,.$$

In total we get

$$g^{\mathrm{cost}}(\mathcal{L}, \pi, \mathcal{R}) := c_1 \cdot \mathrm{dur}_{\mathrm{full}} + c_2 \cdot \mathrm{length}_{\mathrm{full}} + c_3 \cdot \mathrm{veh} + c_4 \cdot \mathrm{dur}_{\mathrm{empty}} + c_5 \cdot \mathrm{length}_{\mathrm{empty}}. \tag{4}$$

Travel times. For determining the travel time we follow the traditional approach of fixing the passengers' routes when constructing the event-activity network, assuming that the passengers use these assigned paths. In the event-activity network, passengers are routed on a shortest path according to the lower bounds on the activities and assigned as weights c_a to the activities $a \in \mathcal{A}$. Additionally to the travel time, we consider a penalty pen for every transfer. The total *perceived* travel time on these fixed paths can then be determined as

$$g^{\text{time}}(\mathcal{L}, \pi, \mathcal{R}) = \sum_{a=(i,j)\in\mathcal{A}} c_a \cdot (L_a + (\pi_j - \pi_i - L_a \mod T)) + \sum_{a\in\mathcal{A}_{trans}} c_a \cdot \text{pen}. \tag{5}$$

Note that the travel time does not depend on the vehicle schedule.

The two objective functions we have sketched here are common in the literature when broken down to one single planning stage:

Nearly all papers dealing with vehicle scheduling minimize a combination of empty kilometers and number of vehicles needed, i.e., $\text{veh} + a \cdot \text{length}_{\text{empty}}$. This is equivalent to g^{cost} if the duration of full and empty rides are weighted equally and a is chosen as $a = \frac{c_5}{c_3}$ since the duration and the length of the lines are all known due to the timetable being fixed.

In timetabling, the goal is usually to minimize the sum of (perceived) travel times for the passengers. Since it is computationally very difficult, most papers make the simplifying assumption that the number of travelers on every activity in the event-activity network is known and fixed, as it is done here.

Pareto optimal LTS-plans. We call a feasible LTS-plan $(\mathcal{L}, \pi, \mathcal{R})$ *Pareto optimal* if there does not exist another LTS-plan $(\mathcal{L}', \pi', \mathcal{R}')$ which satisfies

$$g^{\text{cost}}(\mathcal{L}', \pi', \mathcal{R}') \leq g^{\text{cost}}(\mathcal{L}, \pi, \mathcal{R}), \quad g^{\text{time}}(\mathcal{L}', \pi', \mathcal{R}') \leq g^{\text{time}}(\mathcal{L}, \pi, \mathcal{R})$$

with one of the two inequalities being strict.

3 Traditional sequential approach

The traditional approach is a combination of algorithms which have been described in the literature. It goes through line planning, timetabling, and vehicle scheduling sequentially and finds (close to) optimal solutions in each of the steps.

Step L: Line planning. There exists a variety of algorithms for line planning, see [21]. Some of them assume a line pool to be given, others determine the lines during their execution ([2]). If a line pool is required, a line pool generation procedure can be used (see [7] and references therein).
In our experiments: We use the cost model for a fixed line pool which is either given (dataset Bahn) or generated by [7] (dataset Grid).

Step T: Timetabling. Solving the integer programming formulations is too time-consuming for most instances, hence often heuristics ([9, 15, 16]) are used.
In our experiments: We use the fast MATCH heuristic [16].

Step S: Vehicle scheduling. There exists a variety of algorithms, see [4].
In our experiments: We use the flow-based model of [4].

We remark that even if all three steps are solved optimally, the resulting LTS-plan need not be Pareto optimal. This is due to the sequential approach: the line plan is the basis for the timetable and the vehicle schedule, but optimal lines cannot be determined without knowing the optimal timetable and the optimal vehicle schedule.

4 Look-ahead enhancements

As already mentioned, the vehicle schedules have a large impact on the costs of an LTS-plan. Since the vehicle schedules are determined only in the last of the three considered planning stages, the costs of an LTS-plan determined by the sequential approach are usually not minimal. We propose three enhancements in order to receive LTS-plans with better costs than in the sequential approach. We nevertheless also evaluate the perceived travel times for the passengers.

4.1 Using new costs in the line planning step

When evaluating the costs of an LTS-plan, (4) shows that the costs are determined to a large amount by the number of vehicles needed. Even if as few lines as possible are established it is not clear how many vehicles are needed in the end and how many empty kilometers are necessary.

In the traditional approach the costs of a line are usually assumed to be proportional to its length with some fixed costs to be added, i.e.,

$$\overline{\text{cost}_l} = \text{cost}_{\text{fix}} + c \cdot \text{length}_l \tag{6}$$

where $\text{cost}_{\text{fix}} \in \mathbb{R}_+$ and $c \in \mathbb{R}_+$ is a scaling factor.

Here, we now try to compute the costs of a line as closely as possible to the costs it may have later in the evaluation of the LTS-plan. The idea is to approximate the costs per line by distributing the costs specified in (4) to the lines and computing the costs per period, i.e., we want to get

$$g^{\text{cost}} \approx \sum_{l \in \mathcal{L}} \text{cost}_l \cdot \text{p}.$$

For full duration and distance this can be done straightforwardly, as we only need to know the number of planning periods which are considered in total as the length and duration of a line does not change between periods. Under our assumptions, we know the duration of a line beforehand by (3). The number of vehicles needed, the empty distance and the empty duration are in general more difficult to approximate as they can differ between the planning periods due to an aperiodic vehicle schedule. As upper bound we use a very simple vehicle schedule where all vehicles periodically cover only one line and its backwards direction. This gives us that the empty distance is always zero and can be neglected. The empty duration of a line can be computed as

$$\text{empty duration after driving on line } l = \frac{T}{2} - (\text{dur}_l \mod \frac{T}{2}),$$

and for a given minimal turnaround time $L_{\min}$ of a vehicle, the number of vehicles needed to serve a line and its backwards direction can be approximated by

$$\#\text{vehicles needed for line } l \text{ and backwards direction} = \lceil 2 \cdot (\text{dur}_l + L_{\min})/T \rceil .$$

Summarizing, we can approximate the line costs as:

$$\text{cost}_l = 2 \cdot c_1 \cdot \text{dur}_l + 2 \cdot c_2 \cdot \text{length}_l + \frac{c_3}{\text{p}} \cdot \left\lceil 2 \cdot \frac{\text{dur}_l + L_{\min}}{T} \right\rceil + 2 \cdot c_4 \cdot \left(\frac{T}{2} - \text{dur}_l \mod \frac{T}{2} \right). \tag{7}$$

4.2 Line pool generation with look-ahead

The next idea is to take account of good vehicle schedules already in the very first step: we construct the lines in the line pool in a way such that no empty kilometers are needed and that the resulting lines are likely to be operated with a small number of vehicles.

To create a line pool which already considers the vehicle routing aspect, we modified the line pool generation algorithm described in [7]. For a given minimal turnaround time $L_{\min}$ of a vehicle and a maximal allowed buffer time α we ensure that the duration dur_l as defined in (3) of a line l satisfies

$$\frac{T}{2} - L_{\min} - \alpha \leq \mathrm{dur}_l \mod \frac{T}{2} \leq \frac{T}{2} - L_{\min}. \tag{8}$$

Here, the duration of a line is computed according to the minimal driving time on edges and the minimal waiting time in stops. Equation (8) ensures that at the end of a trip, i.e., the driving of a line, the vehicle has enough time to start the trip belonging to the backwards direction of the same line and has to wait no more than α minutes to do so. Thus, we get that the round-trip of forward and backward direction together differs from an integer multiple of the period length by at most $2 \cdot \alpha$.

4.3 Vehicle scheduling first

In our last suggestion we propose to switch Step T and Step S in the sequential approach, i.e., to find (preliminary) vehicle schedules directly after the line planning phase. This is particularly interesting if the line plan contains lines which can be operated efficiently by one vehicle, i.e., lines with small α, since it ensures that the timetable will not destroy this property. This is done as follows:

Step L: This step is done as in the traditional approach.

S-first: For every line l we introduce *turnaround* activities in the periodic event-activity network between the last arrival event of the line in forward direction and the first departure event of the line in backward direction, and vice versa. The lower bound for these activities is set to $L_{\min}$ and the upper bound to $L_{\min} + 2 \cdot \alpha$. These activities ensure that the timetable to be constructed in the next step allows the vehicle schedule we want, namely that only one vehicle operates the line.

Step T: We then proceed with timetabling as in the traditional approach but respecting the turnaround activities such that the resulting timetable does not destroy the desired vehicle schedule.

Step S: After timetabling we perform an additional vehicle scheduling step as in the classic approach: We delete the turnaround activities and proceed with vehicle scheduling as usual. Nevertheless, it is likely, that many of the vehicle routes already determined in S-first will be found again.

Note that S-first can be performed very efficiently in the number of lines in the line concept. We furthermore remark that for a line plan in which all lines have a buffer time $\alpha = 0$, the Step S can be omitted since having line-pure vehicle schedules is an optimal solution in such a case. Even if not all lines have zero buffer times, fixing a timetable in Step T with respecting the turnaround activities often already determines the optimal vehicle schedule. This means that vehicle scheduling in Step S is often redundant, which was not only observable in most cases of our experiments, but is also illustrated more precisely in Example 1 of the appendix.

5 Experiments

We compared the traditional approach for finding an LTS-plan against the enhancements proposed using LinTim, a software framework for public transport optimization [1, 8]. We use the following parameters to describe the different combinations of our enhancements.

1. Using the new costs (7) in line planning (Step L) as proposed in Section 4.1 is denoted by `new cost`, whereas traditional costs are denoted as `normal cost`.
2. The second option, described in Section 4.2 is to construct a new pool (`new pool`), whereas `normal pool` uses some given (standard) pool for line planning (Step L). Combining both pools has been done in a third option (`combined pool`).
3. The decision of computing the timetable or the vehicle schedules first (so using Step S-first from Section 4.3), is denoted by `TT first` and `VS first` respectively.

As test instances we used two significantly different datasets.

Dataset Grid: A grid graph of 5 by 5 nodes and 40 edges, which is a model for a bus network constructed in [5]. In this example, we have $T = 20$ and we used $\mathrm{p} = 24$ periods. The `normal pool` for this instance has been calculated with the tree based heuristic from [7].

Dataset Bahn: This is a close-to-real world instance which consists of 250 stations and 326 edges describing the German ICE network. The period length is $T = 60$, we computed for $\mathrm{p} = 32$ periods in order to achieve a reasonable time horizon for vehicle scheduling. Note that p is even larger in practical railway applications. As `normal pool` we used a pool of Deutsche Bahn. For the computations we used a standard notebook with i3-2350M processor and 4 GB of RAM. The computation time for one data point of the Grid dataset did not exceed 3 min, while computing a solution for the Bahn dataset took up to 30 minutes.

5.1 Dataset Grid

Figure 1 shows 12 solutions, one for every combination of our parameters. These are graphed according to travel times (x-axis) and their costs (y-axis). We computed the costs and the travel times of the LTS-plans as described in (4) and in (5). We observe the following:

- The solution of the traditional approach (circle with grey marker, left side filled) is dominated by the solution obtained when replacing `normal pool` by `combined pool`.
- Using `new cost` (black markers) instead of `normal cost` (grey markers) always decreases the costs.
- Using `combined pool` always has better costs than using `new pool` or `normal pool`. The travel times sometimes decrease and sometimes increase.
- The option `TT first` yields better travel times compared to `VS first` while `VS first` always has lower costs than `TT first`.
- There are five non-dominated solutions, four of them computed by using `new cost`. Whenever `new pool` or `combined pool` was used together with `new cost` the resulting solution was non-dominated.

The new pool to be generated depends on the parameter α. In Figure 1, $\alpha = 3$ was used. We also tested the parameters $\alpha = 2, 3, \ldots, 10$ for all combinations. The result is depicted in Figure 2. Note that $\alpha \geq 10$ implies no restrictions on the line lengths.

The basic findings described for $\alpha = 3$ remain valid also for other line pools generated: Solutions generated with `new cost` have lower costs while solutions generated with `normal cost` have smaller travel times. The leftmost solutions correspond to `TT first` and bottom-most solutions correspond to `VS first`. In fact, for every single LTS-plan that has been

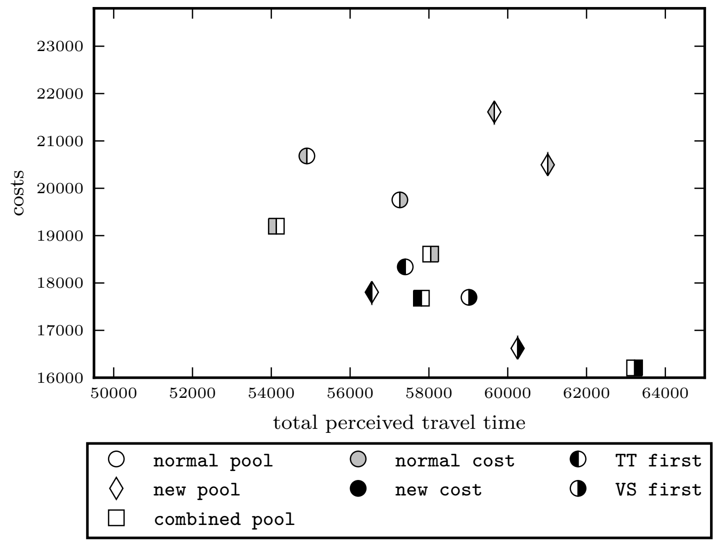

Figure 1 Different combinations of look-ahead steps.

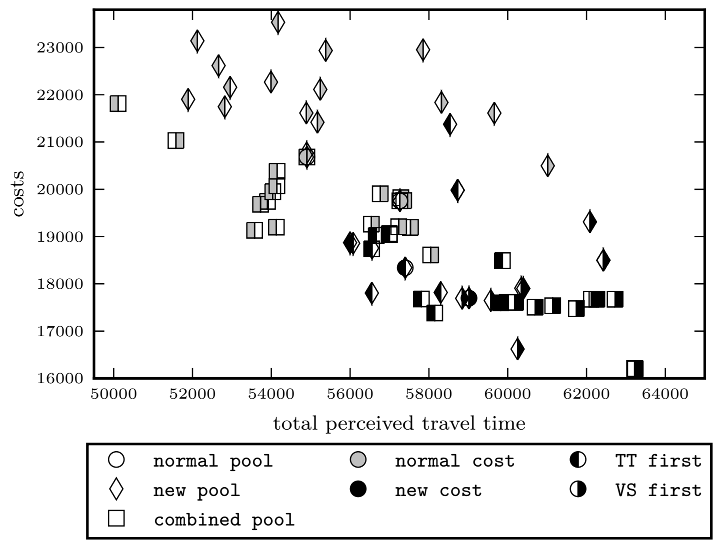

Figure 2 Different combinations of look-ahead steps and different choices for α.

computed, `VS first` yielded a cheaper solution than `TT first` while the latter resulted in a solution with smaller travel time than `VS first`. Finally, none of the solutions computed by using `normal pool` is non-dominated; the Pareto front (i.e., the non-dominated solutions) consists mostly of squares, i.e., solutions generated with `combined pool`. Nevertheless, we see that the quality of the solution obtained depends significantly on the choice of the parameter α. This is investigated in Figure 3.

First of all, we again see that for every fixed α `new cost` yields better solutions than `normal cost` and that the `combined pool` always yields lower costs than `new pool`. If all three look-ahead enhancements `new cost`, `combined pool` and `VS first` are applied, there is a trend of increasing costs once α increases, corresponding to the conjecture that cheap LTS-plans can be found by a small choice of α. For $\alpha = 0$ and $\alpha = 1$ the restrictions on the line length implied by equation 8 is in this example of a grid graph so strict that no feasible solution is possible.

5.2 Dataset Bahn

Applying the implemented enhancements to Bahn with the parameter choice $\alpha = 10$ (Note that $\alpha = 3$ for $T = 20$ in dataset Grid is similar to $\alpha = 10$ for $T = 60$ in dataset Bahn.) yields the results depicted in Figure 4.

The remarkable thing observable in this scenario is that `new` and `combined pool` lead to drastically vehicle cost reductions of more than 40%, whereas the travel time increases by up to 20%. Next to the fact of `combined pool` leading to better costs also the behaviour of `TT first` against `VS first` remains similar to the Grid instance. One can see that `VS first` saves costs between 1 and 5% and `TT first` decreases the travel time by 1 to 3 %. Since the size of the generated line pool had to be chosen small in comparison to the instance size (because of runtime and memory limitations), also the number of feasible line concepts is comparable small. Therefore, this example did not show any impact of using `normal` or `new cost` to the vehicle scheduling costs.

6 Relation to the Eigenmodel

In [22], it is proposed to use different paths through the Eigenmodel (depicted in Figure 5 in the appendix) when optimizing an LTS-plan. In this model, the traditional approach (`normal cost`, `normal pool`, `TT first`) has been depicted as the blue path starting with line planning, then finding a timetable and finally a vehicle schedule. In this paper we compared this traditional approach to two other paths:

- The approach (`normal cost`, `normal pool`, `VS first`) corresponds to the red path in which first a line planning step is performed, then vehicle schedules are determined and finally a timetable. We have seen that this approach leads to significantly better costs but to a higher travel time.
- The approach (`new cost`, `new pool`, `VS first`) can be interpreted as the green path in which we start with vehicle scheduling (by generating a line pool with small α only containing lines with low vehicle scheduling costs), choose a line plan out of this pool and finally determine a timetable which respects the preferred vehicle schedules. In Figure 1 we see that this approach generated the solution with lowest costs. Neglecting the tiny difference between `normal` and `new cost` this also holds for the Bahn instance.

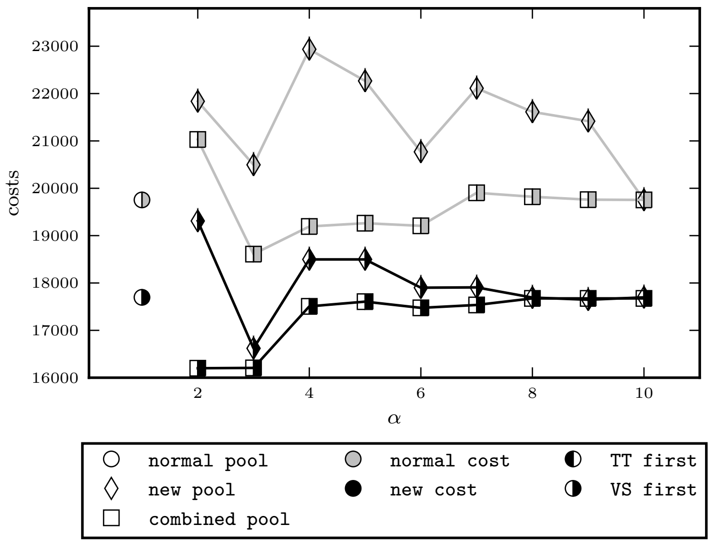

Figure 3 Impact of choice for α.

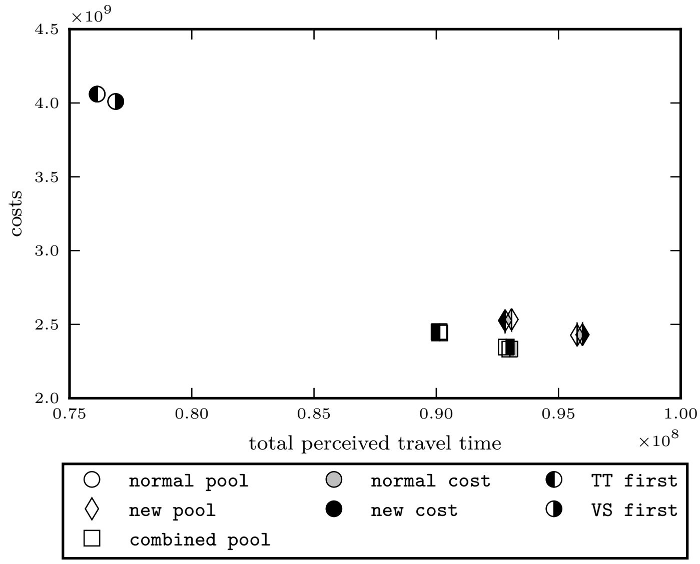

Figure 4 Different combinations of look-ahead steps.

7 Outlook and further research

Summarizing our experiments, all three look-ahead enhancements lead in the majority of cases to a cheaper LTS-plan. Even choosing only one of the approaches will most likely lead to this goal. It is remarkable that the implementation of the proposed algorithmic ideas even performs very well on the Bahn dataset, that has the size and structure of a real world instance. Since exact approaches are far away from solving data sets of this size, the look-ahead heuristic proves itself useful for revealing the strength of considering integrated public transportation optimization.

The presented look-ahead approaches are designed to find a cost-optimized LTS-plan. One could also try to find heuristic approaches focussing on finding a passenger-convenient LTS-plan. A possible step towards this direction would be to choose a different line planning procedure, in order to optimize not with respect to the costs, but for example with respect to the number of direct travelers in the network.

Further research could also be carried out regarding exact approaches of integrated public transportation planning. It would be interesting to investigate different ways of decomposing the integrated problem, in particular, if also routing decisions are included. First results are under research, see [13].

References

1 S. Albert, J. Pätzold, A. Schiewe, P. Schiewe, and A. Schöbel. LinTim – Integrated Optimization in Public Transportation. Homepage. see `http://lintim.math.uni-goettingen.de/`.

2 R. Borndörfer, M. Grötschel, and M. Pfetsch. A Column-Generation Approach to Line Planning in Public Transport. *Transportation Science*, 41:123–132, 2007.

3 R. Borndörfer, H. Hoppmann, and M. Karbstein. Passenger routing for periodic timetable optimization. *Public Transport*, 2016. `doi:10.1007/s12469-016-0132-0`.

4 S. Bunte and N. Kliewer. An overview on vehicle scheduling models. *Public Transport*, 1(4):299–317, 2009.

5 M. Friedrich, M. Hartl, A. Schiewe, and A. Schöbel. Angebotsplanung im öffentlichen Verkehr – planerische und algorithmische Lösungen. In *Heureka'17*, 2017.

6 P. Gattermann, P. Großmann, K. Nachtigall, and A. Schöbel. Integrating Passengers' Routes in Periodic Timetabling: A SAT approach. In Marc Goerigk and Renato Werneck, editors, *16th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2016)*, volume 54 of *OpenAccess Series in Informatics (OASIcs)*, pages 1–15, Dagstuhl, Germany, 2016. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. `doi:10.4230/OASIcs.ATMOS.2016.3`.

7 P. Gattermann, J. Harbering, and A. Schöbel. Line pool generation. *Public Transport*, 9(1):7–32, 2017. `doi:10.1007/s12469-016-0127-x`.

8 M. Goerigk, M. Schachtebeck, and A. Schöbel. Evaluating Line Concepts using Travel Times and Robustness: Simulations with the LinTim toolbox. *Public Transport*, 5(3), 2013.

9 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 S. Harrod. A tutorial on fundamental model structures for railway timetable optimization. *Surveys in Operations Research and Management Science*, 17:85–96, 2012.

11 C. Liebchen. *Periodic Timetable Optimization in Public Transport.* dissertation.de – Verlag im Internet, Berlin, 2006.

12 C. Liebchen and R. Möhring. The Modeling Power of the Periodic Event Scheduling Problem: Railway Timetables – and Beyond. In *Proceedings of 9th meeting on Computer-Aided Scheduling of Public Transport (CASPT 2004)*. Springer, 2004.

13 M. Lübbecke, C. Puchert, P. Schiewe, and A. Schöbel. Detecting structures in network models of integrated traffic planning. Presentation at the Clausthal-Göttingen International Workshop on Simulation Science.

14 K. Nachtigall. *Periodic Network Optimization and Fixed Interval Timetables*. PhD thesis, University of Hildesheim, 1998.

15 K. Nachtigall and J. Opitz. Solving Periodic Timetable Optimisation Problems by Modulo Simplex Calculations. In *Proc. ATMOS*, 2008.

16 J. Pätzold and A. Schöbel. A Matching Approach for Periodic Timetabling. In Marc Goerigk and Renato Werneck, editors, *16th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2016)*, volume 54 of *OpenAccess Series in Informatics (OASIcs)*, pages 1–15, Dagstuhl, Germany, 2016. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. `doi:10.4230/OASIcs.ATMOS.2016.1`.

17 L. Peeters. *Cyclic Railway Timetabling Optimization*. PhD thesis, ERIM, Rotterdam School of Management, 2003.

18 L. Peeters and L. Kroon. A Cycle Based Optimization Model for the Cyclic Railway Timetabling Problem. In S. Voß and J. Daduna, editors, *Computer-Aided Transit Scheduling*, volume 505 of *Lecture Notes in Economics and Mathematical systems*, pages 275–296. Springer, 2001.

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

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

21 A. Schöbel. Line planning in public transportation: models and methods. *OR Spectrum*, 34(3):491–510, 2012.

22 A. Schöbel. An Eigenmodel for Iterative Line Planning, Timetabling and Vehicle Scheduling in Public Transportation. *Transportation Research C*, 74:348–365, 2017. `doi:10.1016/j.trc.2016.11.018`.

23 P. Serafini and W. Ukovich. A mathematical model for periodic scheduling problems. *SIAM Journal on Discrete Mathematic*, 2:550–581, 1989.

A Appendix

The following example shows that it is unlikely to find a better vehicle schedule in Step S.

▶ **Example 1.** Consider two lines l_1 and l_2 such that line l_1 ends at the station that l_2 starts at as shown in Figure 6.

Let the duration of the lines be $\mathrm{dur}_{l_1} = \frac{T}{2}+\epsilon$ and $\mathrm{dur}_{l_2} = \frac{T}{2}-\epsilon$ such that $\mathrm{dur}_{l_1}+\mathrm{dur}_{l_2} = T$. Then using S-first with $L_{\min} = 0$ we will need two vehicles vehicles to serve line l_1 and an additional vehicle to serve line l_2, as the following computation shows. The corresponding vehicle schedule can be seen in Figure 7.

$$\left\lceil \frac{2\cdot(\frac{T}{2}+\epsilon)}{T} \right\rceil = \left\lceil \frac{T+2\cdot\epsilon}{T} \right\rceil = 2$$

$$\left\lceil \frac{2\cdot(\frac{T}{2}-\epsilon)}{T} \right\rceil = \left\lceil \frac{T-2\cdot\epsilon}{T} \right\rceil = 1$$

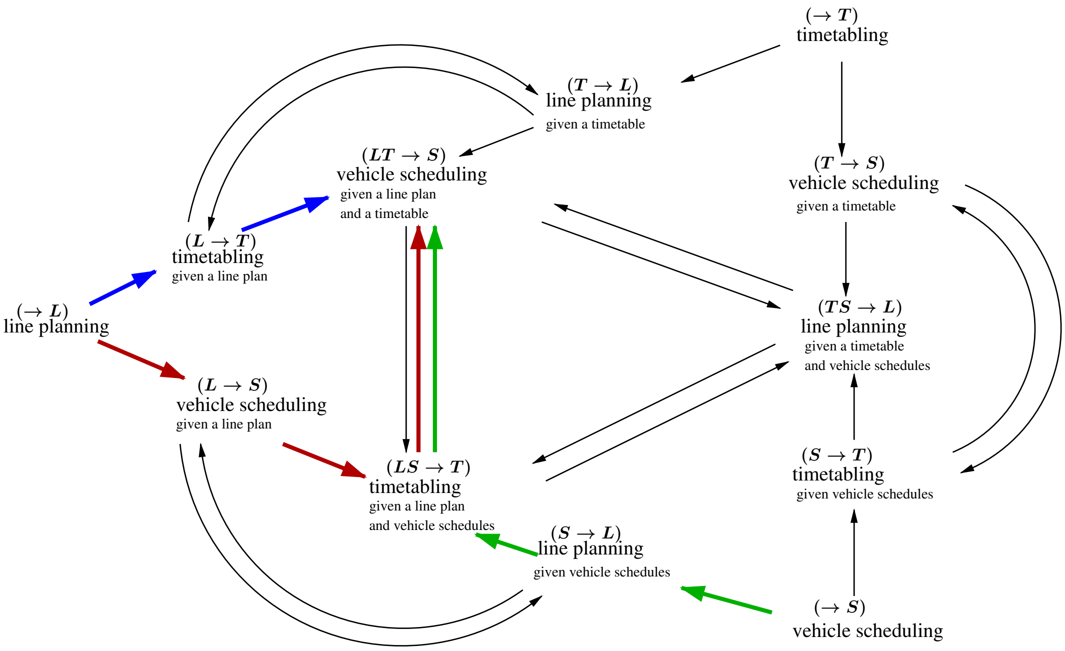

Figure 5 The paths investigated in the Eigenmodel.

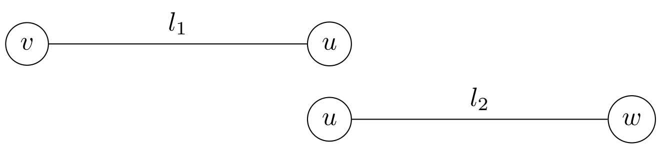

Figure 6 Lines overlapping at station u.

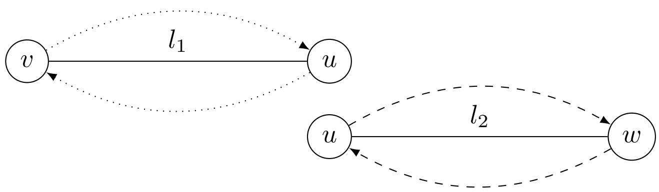

Figure 7 Vehicle schedule derived by S-first.

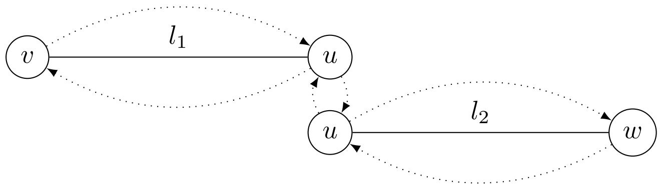

Figure 8 Optimal vehicle schedule.

However, both lines could also be served consecutively by the same vehicle, leading to a total of two instead of three vehicles as can be seen in Figure 8.

$$\left\lceil \frac{2 \cdot (\frac{T}{2} + \epsilon + \frac{T}{2} - \epsilon)}{T} \right\rceil = \left\lceil \frac{2 \cdot T}{T} \right\rceil = 2\,.$$

Nevertheless, it is very unlikely that this vehicle schedule is possible after the timetabling stage T. Consider an OD-pair from v to w. These passengers have to transfer at station u with a minimal transfer time of $\epsilon' > 0$. Then, during the timetabling stage (Step T), the lines will be synchronized such that the passengers can transfer at station u. Therefore, the vehicle schedule shown in Figure 8 will also need three vehicles:

$$\left\lceil \frac{2 \cdot (\frac{T}{2} + \epsilon + \frac{T}{2} - \epsilon + \epsilon')}{T} \right\rceil = \left\lceil \frac{2 \cdot T + 2 \cdot \epsilon'}{T} \right\rceil = 3\,.$$

This shows that the vehicle schedule computed in Step S-first is already optimal as the vehicle schedule shown in Figure 7 is still feasible.