Timetabling with Passenger Routing

M. Schmidt, A. Schöbel

Nr. 24
Timetabling with Passenger Routing *

Marie Schmidt and Anita Schöbel
Institute for Numerical and Applied Mathematics
University of Göttingen
Email: {m.schmidt,schoebel}@math.uni-goettingen.de

3rd September 2013

Abstract

Customer-oriented optimization of public transport needs data about the passengers in order to obtain realistic models. Current models take passengers' data into account by using the following two-phase approach: In a first phase, routes for the passengers are determined. In a second phase, the actual planning of lines, timetables, etc. takes place using the knowledge on which routes passengers want to travel from the results of the first phase. However, the actual route a passenger will take strongly depends on the timetable, which is not yet known in the first phase. Hence, the two-phase approach finds non-optimal solutions in many cases. In this paper we study the integrated problem of determining a timetable and the passengers' routes simultaneously. We investigate the computational complexity of the problem and present solution approaches which are tested on close-to real world data.

1 Introduction

Public transportation planning covers a wide range of decision problems which arise when planning and operating public transportation systems. Besides long-term decisions concerning the network design itself many decision problems in public transportation arise on a strategic level, e.g., the design of line plans, timetables, and of vehicle and crew schedules. Furthermore, in case of disturbances, disposition problems have to be solved to update timetables and schedules in order to minimize the impact of the disturbances on the performance of the system.

In this paper we consider the timetabling problem, i.e. our goal is to determine the arrival and departure times of the vehicles. Our results can be applied to many different public transportation systems like train, bus or metro systems. However, for the sake of simplicity, in this paper we use the terminology from rail transportation to explain our concepts and results.

More precisely, given a public transportation network, trains operating on given lines in this network, and required transfer possibilities between the trains, the task of timetabling is to assign a point in time to every departure and every arrival of a train at a station. Our objective is to minimize the passengers' total travel time.

In order to minimize the passengers' travel time, data about their travel demand is needed. Such data is usually given as a set of origin-destination pairs (OD-pairs) specifying from which station to which other station passengers want to travel. Current timetabling models (see, e.g., [15, 18, 22, 25, 27]) take the OD-pairs into account by using a first phase (routing phase) as preprocessing in which a route for every OD-pair is computed. This can be done by determining a shortest path in the public transportation network with respect to the physical edge lengths, or by more sophisticated traffic assignment procedures. When the passengers' routes have been determined

---

*This work was partially supported by the Marie Curie Action International Research Staff Exchange Scheme (IRSES) FP7-PEOPLE-2009-IRSES project OptALI under project ID 246647*
it is known how many passengers use which train or which transfer. This knowledge is then used in the second phase (timetabling phase) in which the actual planning takes place and a timetable is determined which is best for the passengers if they travel as assumed in the routing phase. However, in most cases passengers are indifferent to the stopovers on their routes. I.e., if there are several alternative routes from a passenger’s origin to the destination, the passenger will consult the current timetable and choose the route with shortest travel time accordingly. In this case, different timetables may yield different routes. Hence, the passengers’ route choice depends on the timetable. On the other hand, when minimizing total travel time, the design of the timetable depends on passengers’ route choice. On that account, the two-phase approach described above in general leads to suboptimal results with respect to the total travel time. For this reason, in this paper we introduce a model where the timetable and the passengers’ routes are determined simultaneously, i.e., we integrate the routing and the timetabling phase. We call this model timetabling with routing (TTwR).

Timetabling is distinguished into periodic and aperiodic timetabling. While in periodic timetabling it is assumed that departures and arrivals are repeated after a certain time period, in aperiodic timetabling every train is scheduled individually and departures and arrivals are not required to follow a periodic pattern. In terms of computational complexity, aperiodic timetabling is much easier than periodic timetabling: e.g., in periodic timetabling it is even NP-hard to find a feasible timetable [38, 22], while this can be done in polynomial time for aperiodic timetabling [29]. For this reason, in this paper we restrict ourselves to aperiodic timetabling to analyze the effect of integrating the passenger routing on the complexity of the timetabling problem. The relatively easier problem allows us to study how far the integration of timetabling and passenger routing can be driven.

We show that TTwR is strongly NP-hard and hard to approximate and thus rule out the possibility of finding a polynomial-time exact algorithm (or one with a good approximation ratio) unless \( P = NP \). We then investigate a simplified problem where a passenger does not only specify his/her origin and destination but also the first train he/she wants to board and the train he/she wants to arrive with. This problem is much easier than TTwR and can be solved by linear programming. Based on this result, we develop two exact solution approaches with non-polynomial running time for TTwR. Numerical tests on close to real-world data show that our approach is able to find an optimal solution to TTwR in reasonable running time for medium-sized instances in most cases.

The remainder of the paper is structured as follows. We start with an overview on related literature in Section 2. In Section 3 we describe the model of timetabling with fixed passenger routes which is commonly used in the timetabling literature. To this end, we describe the underlying event-activity network and state the well-known formulation of this problem as a linear program. In Section 4 we introduce our new model timetabling with routing (TTwR). The complexity of this problem is analyzed in Section 5. In Section 6 we show that a simplified version of the problem is solvable in polynomial time and develop two solution approaches based on this result. Finally, in Section 7 we test the integer programming approach and compare it to an alternative IP formulation.

2 Literature overview

As done in most papers (see, e.g., [15, 18, 22, 25, 27]), we model timetabling problems using event-activity networks (EANs). This concept relies on the periodic event-scheduling (PESP) framework developed by [38]. Many recent studies deal with the computationally challenging problem of periodic timetabling since [38] have shown that finding feasible periodic timetables is strongly NP-complete. This result even holds for a fixed length of the time period [26]. Integer programming formulations for periodic timetabling have been presented in, e.g., [15, 17, 18, 22, 25, 27]), heuristic solution methods are suggested in [11, 23].

The problem of finding a feasible aperiodic timetable is discussed in [29] under the name of feasible differential problem. There, a feasibility criterion and a polynomial-time solution algorithm for the
feasible differential problem are developed. Since aperiodic timetabling is easily solvable, not much literature is devoted to the pure problem. It appears as a subproblem in delay management where an aperiodic disposition timetable is to be found, see [34] for a summary of three approaches for finding such an aperiodic timetable. A more challenging line of research is aperiodic timetabling with capacity constraints which turns out to be an NP-complete problem. A survey on aperiodic timetabling with capacity constraints is [21], aperiodic rescheduling of a timetable under capacity constraints in case of disturbances has been treated in [14]. The framework of aperiodic timetabling is also used to develop, test and evaluate new concepts, like for example in [8, 10, 16, 36] where robustness approaches are tested on the aperiodic timetabling problem.

Almost all models in the assume that passengers’ routes are fixed before the design of a timetable starts. There are only a few exceptions: Nachtigall [22] remarks that passenger routes depend on the choice of the timetable and discusses one chapter to a periodic timetabling model with integrated passenger routing. He proposes an iterative heuristic to solve the problem and briefly covers special cases where the problem is solvable in polynomial time. The subject has recently gained more attention and has also been considered in some diploma theses [1, 13, 20, 39]. The importance of integrating routing decisions into the design process in public transportation has also been mentioned in [33] where a first analysis for the timetabling case is provided.

The integration of passengers’ routing decisions into the optimization process has recently become common in line planning. See, e.g., [3, 4, 28, 31, 33, 35, 37]) for integer programming approaches and [30] for a systematic list of results on the computational complexity of line planning with routing. In delay management, the complexity of integrating routing decisions has been studied in [6, 30, 32], solution approaches have been proposed and tested in [5, 6].

3 Timetabling with fixed passenger routes

In this section we briefly summarize how aperiodic timetabling is modeled in the literature. In order to distinguish this problem from our new model we call it timetabling with fixed passenger routes (TTF). We first describe how to model TTF in an event-activity network and state a well-known linear programming formulation for this problem.

In TTF we are given a public transportation network (PTN) $G = (S, E)$, consisting of stations $S$ and direct connections $E \subseteq S \times S$. We are given a line plan specifying on which paths through the PTN trains operate. Furthermore, it is specified in the input between which trains and at which stations transfers should be possible. To this end, we are given a set $T$ of required transfer activities, the movement of passengers from one train to another. We define

- the set of departure events $E_{\text{dep}} := \{(tr - s - \text{Dep}) : tr \in TR, s \in S(tr)\}$ and
- the set of arrival events $E_{\text{arr}} := \{(tr - s - \text{Arr}) : tr \in TR, s \in S(tr)\}$.

The set of operational events is $E_{\text{op}} := E_{\text{dep}} \cup E_{\text{arr}}$.

The arcs of the EAN are called activities since they represent physical activities of the trains, i.e., driving between stations and waiting at stations to let passengers get on and off, or, in case of the transfer activities, the movement of passengers from one train to another. We define

- the set of driving activities $A_{\text{drive}} \subseteq E_{\text{arr}} \times E_{\text{dep}}$ as
  $A_{\text{drive}} := \{(tr - s - \text{Dep}, tr - s' - \text{Arr}) : tr \in TR, (s, s') \in E(tr)\}$,
- the set of waiting activities $A_{\text{wait}} \subseteq E_{\text{arr}} \times E_{\text{dep}}$ as
  $A_{\text{wait}} := \{(tr - s - \text{Arr}, tr - s - \text{Dep}) : tr \in TR, s \in E(tr)\}$,
Figure 1: Cutout of an EAN: the nodes represent arrivals (‘Arr’) and departures (‘Dep’). The continuous arcs are driving and waiting activities, the dashed arcs represent transfer activities.

- the set of transfer activities \( A_{\text{transfer}} \subset E_{\text{arr}} \times E_{\text{dep}} \) as

\[
A_{\text{transfer}} := \{(tr - s - Arr, tr' - s - Dep) : (tr, tr', s) \in C\}
\]

and denote by \( A_{\text{op}} := A_{\text{drive}} \cup A_{\text{wait}} \cup A_{\text{transfer}} \) the set of operational activities.

If \( \pi_i \) denotes the time of event \( i \in E_{\text{op}} \), and \( a = (i, j) \in A_{\text{op}} \) an activity linking event \( i \) and event \( j \), a timetable \( (\pi_i)_{i \in E_{\text{op}}} \) is feasible if every activity \( a = (i, j) \in A_{\text{op}} \) satisfies

\[
l_a = l_{(i, j)} - \pi_j + \pi_i \leq u_{(i, j)} = u_a
\]

for some given lower and upper bounds \( l_a \) and \( u_a \) on the duration of activity \( a \).

In TTF, passenger routes have been calculated in a first (preprocessing) phase, e.g., by using shortest-path algorithms in the EAN with some approximation of the traveling time or by more sophisticated traffic assignment procedures yielding a set of paths \( P \) and a number of passengers \( w_P \) for every path \( P \in \mathcal{P} \). The number of passengers using an activity \( a \in A_{\text{op}} \) (activity weight) can then be calculated as

\[
w_a := \sum_{P \in \mathcal{P}, P \ni a} w_P.
\]

Since for a given timetable \( \pi \) the duration of each activity \( (i, j) \in A_{\text{op}} \) is given as \( \pi_j - \pi_i \), the travel time of a passenger on path \( P = (i_1, \ldots, i_Q) \) passing through the events \( i_1, \ldots, i_Q \) is

\[
c_{\text{TTF}}(\pi, P) := \pi_{i_Q} - \pi_{i_1} = \sum_{(i, j) \in P} \pi_j - \pi_i.
\]

Summing up over all passengers we obtain the total travel time

\[
c_{\text{TTF}}(\pi) = \sum_{(i, j) \in A_{\text{op}}} w_{(i, j)}(\pi_j - \pi_i).
\]

Given the EAN \( \mathcal{N} \), upper and lower bounds \( u_a \) and \( l_a \) and activity weights \( w_a \) for every \( a \in A_{\text{op}} \), TTF can be formulated as the following linear program

\[
\min \sum_{(i, j) \in A_{\text{op}}} w_{(i, j)}(\pi_j - \pi_i)
\]

s.t. \( \pi_j - \pi_i \leq u_{(i, j)} \) \quad \forall (i, j) \in A_{\text{op}}

\[
\pi_j - \pi_i \geq l_{(i, j)} \quad \forall (i, j) \in A_{\text{op}}
\]

\[
\pi_i \in \mathbb{R} \quad \forall i \in E_{\text{op}}.
\]

4 Timetabling with integrated routing

In TTF passengers’ routes are considered as input to the timetabling problem. However, under the assumption that the main criterion for passengers’ route choice is travel time, this leads to suboptimal solutions, since the travel time on a path \( P \) depends on the timetable in the way specified in (2) and hence the travel-time minimal route for a passenger cannot be predicted before a timetable is fixed.
For this reason, we consider timetabling with integrated routing (TTwR). In TTwR we aim at determining an optimal pair \((\pi, R)\) of a timetable \(\pi\) and a passenger routing \(R\), i.e., we want to find a timetable and the shortest paths for the passengers simultaneously. In contrast to TTF, passengers’ demand is not given as a set of paths but as a set of origin-destination pairs (OD-pairs) \((u,v) \subset S \times S\) specifying origin and destination of the passengers together with a weight \(w_{uv}\), which represents the number of passengers who want to travel from \(u\) to \(v\) in the considered time period.

A naive approach to solve TTwR could be to enumerate all possible combinations of routes consisting of a path from a departure event at station \(u\) to an arrival event at station \(v\) for each OD-pair, solve TTF for each of these routings and, among these, choose a solution with the best objective value. Of course this approach would only be viable in case of very simple networks and few OD-pairs, since even for one OD-pair \((u,v)\) there could be an exponential number of routes.

Indeed, we will see in Theorem 5.1 that TTwR in general is strongly NP-hard.

To include route choices in our timetabling model, we extend the EAN defined in Section 3. For every OD-pair \((u,v)\) we add an origin node \(u - \text{org}\) and a destination node \(v - \text{dest}\).

\[
E_{\text{org}} = \{ u - \text{org} : \text{there exists } v \in S : (u,v) \in \mathcal{OD} \}, \quad \text{and} \\
E_{\text{dest}} = \{ v - \text{dest} : \text{there exists } u \in S : (u,v) \in \mathcal{OD} \}
\]

denote the respective sets of nodes. These nodes are connected to \((E_{\text{op}}, A_{\text{op}})\) by means of

- **origin activities**

  \[
  A_{\text{org}} := \{ (u - \text{org}, tr - u - \text{Dep}) : u - \text{org} \in E_{\text{org}}, \\
  \text{and } tr \text{ such that } (tr - u - \text{Dep}) \in E_{\text{dep}} \}
  \]

- **destination activities**

  \[
  A_{\text{dest}} := \{ (tr - v - \text{Arr}, v - \text{dest}) : v - \text{dest} \in E_{\text{dest}}, \\
  \text{and } tr \text{ such that } (tr - v - \text{Arr}) \in E_{\text{arr}} \}
  \]

We set \(E := E_{\text{op}} \cup E_{\text{org}} \cup E_{\text{dest}}\) and \(A := A_{\text{op}} \cup A_{\text{org}} \cup A_{\text{dest}}\).

Now every path in the event-activity network \(N := (E, A)\) from \(u - \text{org}\) to \(v - \text{dest}\) represents a possible route for the OD-pair \((u,v)\).

We call a set of paths \(R := \{ P_{uv} : (u,v) \in \mathcal{OD} \}\), where \(P_{uv}\) is a path from \(u - \text{org}\) to \(v - \text{dest}\) in \(N\), a **routing**. Analogously to (2) for a given timetable \(\pi\), the length of a path \(P_{uv} = (u - \text{org}, i_1, i_2, \ldots, i_m, v - \text{dest})\) can be calculated as

\[
c_{TTwR}(\pi, P_{uv}) := \sum_{j=1}^{m-1} \pi_{i_{j+1}} - \pi_{i_j}.
\]

For a given pair \((\pi, R)\) of a timetable and a routing, the total travel time is given as

\[
c_{TTwR}(\pi, R) := \sum_{(u,v) \in \mathcal{OD}} w_{uv} c_{TTwR}(\pi, P_{uv})
\]

TTwR can now be defined as follows: Given an EAN \(\mathcal{N}\) with upper and lower bounds and a set of OD-pairs \(\mathcal{OD}\), find a routing \(R\) and a timetable \(\pi\) such that \(c_{TTwR}(\pi, R)\) is minimized.
5 Computational complexity of timetabling with routing

As we have seen in Section 3, TTF can be solved by linear programming and is hence in the class $P$ of polynomially solvable problems. Note that this even holds if the timetable is required to consist of integer numbers (e.g., minutes) since in this case the upper and lower bounds can be rounded to integer numbers and the constraint matrix is the node-arc incidence matrix of a network and hence totally unimodular (see, e.g., [24]).

In the following we show that in contrast, TTwR is strongly NP-hard and cannot be approximated within a constant factor in polynomial time.

**Theorem 5.1.** The decision version of TTwR is strongly NP-complete (hence TTwR is strongly NP-hard) even if all OD-pairs have the same origin.

**Proof.** We show this by reduction from the strongly NP-complete decision problem Set Cover ([9]) defined as follows: Given $P = \{p_1, \ldots, p_m\}$, a set of subsets $Q = \{q^1, \ldots, q^n\}$ with $q^i \subset P$ and a natural number $K$, decide whether there is a subset $Q' \subset Q$ with $|Q'| \leq K$ such that for every $p_i \in P$ there is at least one $q^i \in Q'$ such that $p_i$ is contained in $q^i$.

Let $(P, Q, K)$ be an instance of Set Cover with $P = \{p_1, \ldots, p_m\}$ and $Q = \{q^1, \ldots, q^n\}$. Without loss of generality we assume that $K < n$. Based on $P$ and $Q$ we construct an instance of TTwR in the following way:

We represent the elements $p_i \in P$ by OD-pairs $(u, v_i)$ with passenger numbers $w_{uv_i} := K + 1$.

The sets $q^i \in Q$ are represented by a gadget consisting of three trains $(tr^j)^1, (tr^j)^2$ and $(tr^j)^3$, six stations $s_1, s_2, s_3, s_4, s_5, s_6$ (where $s_1$ is the same station for all OD-pairs) and an OD-pair $(u, v^j)$ with $u := s_1$ and $v^j := s_3$ and $w_{uv^j} := 1$, in the way depicted in Figure 2.

The square nodes are the departure and arrival events where the train names are omitted in the node labels for the sake of a compact representation. The origin and destination events are represented by ovals. The dotted lines are the origin and destination activities, the solid lines represent driving and waiting activities, transfer activities are represented by dashed lines. The gray line indicates where it is possible to leave this gadget.

![Figure 2: The gadget $g^j$ representing a set $q^j$ in the reduction from Set Cover to TTwR in the proof of Theorem 5.1.](image)

For every $p_i \in q^j$ we introduce a train $tr_i^j$ running from $s_3$ to a station $v_i$ and connect $((tr^j)^2 - s_3^3 - Arr)$ to a departure event $(tr_i^j - s_3^3 - Dep)$ by a transfer activity with upper and lower bound $u_{((tr^j)^2-s_3^3-Arr),(tr_i^j-s_3^3-Dep)} := 0$ and $l_{((tr^j)^2-s_3^3-Arr),(tr_i^j-s_3^3-Dep)} := 0$. $(tr_i^j - s_3^3 - Dep)$ is connected to the arrival event of train $tr_i^j$ in $v_i$ by a driving activity $a$ with upper and lower bound $l_a := 0, u_a := 0$.

See Figure 3 (in combination with Figure 2) for an example of the construction for an instance of Set Cover with $P = \{1, 2, 3, 4\}$ and $Q = \{(2, 3, 4), (1, 4), (2, 3)\}$. The square nodes are the departure and arrival events. The origin and destination events are represented by ovals. The dotted lines are the origin
and destination activities, the solid lines represent driving and waiting activities, transfer activities are represented by dashed lines. The nodes $g^j$ represent the gadgets from Figure 2.

Summarizing, our instance $I$ of TTwR is defined by the following input, which defines an EAN $N$ with upper and lower bounds $l, u$ and a set of OD-pairs $\mathcal{OD}$:

- a set of stations $S = \{s_1\} \cup \{s'_2, s'_3, s'_4, s'_5 : j = 1, \ldots, n\} \cup \{v_i : i = 1, \ldots, m\}$,
- a set of trains $\mathcal{T} = \{(tr^j)^1, (tr^j)^2, (tr^j)^3 : j = 1, \ldots, n\} \cup \{tr^j_i : p_i \in g^j\}$ where $(tr^j)^1$ run on the paths $(s'_2, s'_5), (tr^j)^2$ run on $(s_1, s'_2, s'_3, s'_4, s'_5), (tr^j)^3$ run on $(s_1, s'_4)$ and $tr^j_i$ run on $(s'_3, v_i)$.
- transfers
  \[
  C = \{(tr^j)^2, (tr^j)^1, (tr^j)^2, s'_2, (tr^j)^3, (tr^j)^2, s'_4 : j = 1, \ldots, n\}
  \]
  \[
  \cup \{(tr^j)^2, tr^j_i, s'_5 : p_i \in g^j\}
  \]
- a lower bound $l_{((tr^j)^1 - s'_2 - \text{Dep}, (tr^j)^1 - s'_3 - \text{Arr})} = 1$ on driving activities $((tr^j)^1 - s'_2 - \text{Dep}, (tr^j)^1 - s'_3 - \text{Arr})$ for $j = 1, \ldots, n$ and lower bounds $l_u := 0$ on all other activities. The upper bounds are $u_n = 1$ for
  \[
  a \in \{((tr^j)^1 - s'_2 - \text{Dep}, (tr^j)^1 - s'_3 - \text{Arr}),
  ((tr^j)^2 - s'_2 - \text{Dep}, (tr^j)^2 - s'_3 - \text{Arr}),
  ((tr^j)^2 - s'_4 - \text{Dep}, (tr^j)^2 - s'_3 - \text{Arr}) : j = 1, \ldots, n\}
  \]
- $u_n := 0$ otherwise.
- The set of OD-pairs is $\mathcal{OD} = \{(u, v^j) : j = 1, \ldots, n\} \cup \{(u, v_i) : i = 1, \ldots, m\}$ with $u := s_1$ and $v^j := s'_3$. The weights are $w_{uv^j} = 1$ and $w_{uv_i} = K + 1$.

We show that there is a solution $(\pi, R)$ to the constructed instance $I$ with $c_{\text{TTwR}}(\pi, R) \leq K$ if and only if there is a subset $Q'$ solving the Set Cover problem. We observe that if for an OD-pair $(u, v_i)$ there is a gadget $g^j$ such that $u - \text{org}$ is connected to $g^j$ and there is a length of 0 assigned to $((tr^j)^2 - s'_2 - \text{Dep}, (tr^j)^2 - s'_3 - \text{Arr})$, the OD-pair will
arrive at its destination in time 0 while if there is no such structure, there will be a contribution of $n + 1$ to the objective function. Thus, in a feasible solution, for every OD-pair $(u, v)$ there must be at least one gadget $g_j$ such that $(tr_j^1 - s_j^1 - Dep)$ is connected to $g_j'$ and there is a length of 0 assigned to $((tr_j^2)^2 - s_j^2 - Dep, (tr_j^3)^2 - s_j^3 - Arr)$. Note that in this case it follows that the length of $((tr_j^1)^2 - s_j^1 - Dep, (tr_j^2)^2 - s_j^2 - Arr)$ is 1, hence $c_{TTWTR}(\pi, P') = 1$ for the corresponding timetable $\pi$ and any path $P'$ chosen for OD-pair $(u, v')$.

We conclude that $Q'$ is a solution to the considered instance of Set Cover if and only if assigning a length of 1 to $((tr_j^1)^2 - s_j^1 - Dep, (tr_j^2)^2 - s_j^2 - Arr)$ for all $j$ with $g_j \in C'$ and a length of 0 otherwise leads to a solution to the constructed instance of TTwR with solution value $\leq K$. \qed

There exist a lot of negative results concerning the approximability of Set Cover, see e.g. [2, 7] and references therein. Due to the one-to-one correspondence between solutions and solution values of the optimization version of Set Cover and the instances of TTwR constructed in the proof of Theorem 5.1, these results can easily be transferred to TTwR.

In particular, the fact that there is no constant-factor polynomial-time approximation algorithm for Set Cover unless $P = NP$ [2] implies that there is no constant-factor polynomial-time approximation algorithm for TTwR unless $P = NP$. We refer to [30] for a detailed proof.

6 Solution approaches for timetabling with routing

We now turn our attention to solution approaches for TTwR. We start by looking at a modification of the problem in which OD-pairs are not given as pairs of stations in the PTN but as pairs of events in the event-activity network $E$. It will turn out that this simplified version is easy to solve. Based on this result we develop two algorithmic approaches for TTwR.

6.1 A modification: Timetabling with routing between events

Let us for a moment assume that, instead of knowing only origin and destination of the passengers, we are also given information about the specific trains a passenger wants to board at his/her origin and to arrive with at his/her destination. I.e., we consider a timetabling problem with routing of the problem in which OD-pairs are not given as pairs of stations in the PTN but as pairs of events (TTE).

This (on a first glance small) modification makes the problem much easier to solve which is due to the following observation: If we know the chosen events at both origin and destination, then we also know the total travel time, regardless of which route the passenger takes in between. This explains the big difference to just knowing the stations instead of the events.

Lemma 6.1. Let $N$ be an EAN, $s$ and $t$ be two operational events in $N$ and $P_1, P_2$ be two $s$-$t$-paths. Then for any feasible timetable $\pi$ we have

$$c_{TTF}(\pi, P_1) = c_{TTF}(\pi, P_2).$$

Proof. Denote $P_1 := (s, i_1, i_2, \ldots, i_m, t)$ and $P_2 := (s, j_1, j_2, \ldots, j_n, t)$. Then

$$c_{TTF}(\pi, P_1) = \sum_{(i,j) \in P_1} (\pi_j - \pi_i) = \pi_t - \pi_s = \sum_{(i,j) \in P_2} (\pi_j - \pi_i) = c_{TTF}(\pi, P_2).$$

Hence, given a set of OD-pairs $OD_{op} \subset E \times E$, for every $(s, t) \in OD_{op}$ we can choose an arbitrary $s$-$t$-path as a representative, calculate the activity weights $w_a$ as specified in (1) for this set of paths $P$ and solve (3-5). We can also use the following reformulation.
Corollary 6.2. The linear program given as

\[
\min \sum_{(s,t) \in OD_{op}} w_{st} (\pi_t - \pi_s)
\]

s.t. (4), (5), (6)

solves TTE. In particular, TTE is polynomially solvable.

This reformulation can be interpreted as solving TTF in an extended network which consists of \(N_{op}\) and additional virtual activities \(A_{virt} := \{(s,t) : (s,t) \in OD_{op}\}\) with \(l_a := 0\) and \(u_a := \infty\) for all \(a \in A_{virt}\).

![Figure 4: Example for the implementation of virtual activities](image)

See Figure 4 for an example for the concept of virtual activities: The OD-pair \((tr_1 - s_1 - Dep, tr_2 - s_4 - Arr)\) is represented by the dotted virtual activity with assigned weight \(w_{(tr_1 - s_1 - Dep, tr_2 - s_4 - Arr)}\). Note that three different paths in the EAN can be chosen for operational OD-pair \((tr_1 - s_1 - Dep, tr_2 - s_4 - Arr)\), but that the travel time for a given timetable does not depend on this choice.

We conclude that, in contrast to TTwR, TTE is solvable in polynomial time.

6.2 Timetabling with routing

Using the results of the previous section, we can enhance the naive enumeration approach for TTwR that was sketched at the beginning of Section 4 by enumerating the fewer number of all virtual routings which are defined below. Let \(\mathcal{OD} := \{(u_k, v_k) : k = 1, \ldots, n\} \subset S \times S\) be the set of OD-pairs. For one OD-pair \((u_k, v_k)\) we define

\[
A_{virt}^k := \{(i,j) : (u_k - \text{org}, i) \in \mathcal{E}_{\text{org}}, (j, v_k - \text{dest}) \in \mathcal{E}_{\text{dest}}\}
\]

as the set of virtual activities for \((u_k, v_k)\). Note that although the number of paths between origin node \(u_k - \text{org}\) and destination node \(v_k - \text{dest}\) of an OD-pair \((u_k, v_k)\) may be exponential, the number of possible virtual activities is in \(O(|E|^2)\), and each of these activities represents a set of routes from \(u_k - \text{org}\) to \(v_k - \text{dest}\). Any \(n\)-tuple which consists of one element of \(A_{virt}^k\) for every OD-pair \(k = 1, \ldots, n\) hence represents a routing, and is therefore called a virtual routing.

The set of all possible virtual routings is hence given as the Cartesian product of all sets of virtual activities for the OD-pairs, i.e.

\[
R_{virt} := \{(i^1, j^1), (i^2, j^2), \ldots, (i^n, j^n) : (i^k, j^k) \in A_{virt}^k\}
\]

The number of all virtual routings is in \(O(|E|^{2|\mathcal{OD}|})\).

Consider Figure 5 for an example. We observe that there are five possible paths from \(s_1 - \text{org}\) to \(s_4 - \text{dest}\). However, these paths can be represented using only three virtual activities \(a_1, a_2\) and \(a_3\). Hence, we can solve TTwR by solving three instances of TTE and comparing the objective values.
Figure 5: Solving TTwR by solving TTE: virtual activities for OD-pair \((s_1, s_4)\).

An instance of TTwR can be solved as follows: For every virtual routing \(R\) in \(\mathcal{R}_{\text{virt}}\) solve

\[
\min \sum_{k=1}^{\vert \mathcal{OD} \vert} w_{u_k,v_k}(\pi_{j^k} - \pi_{i^k})
\]

s.t. (4), (5), (6)

according to Corollary 6.2 to determine the best possible timetable \(\pi^*(R)\) for the virtual routing \(R\). Take the solution \((\pi^*(R), R)\) with the best objective value as optimal solution for TTwR.

Using this algorithm we obtain a polynomial-time algorithm in the following cases:

**Theorem 6.3.** TTwR is polynomially solvable in the following cases.

1. If for every \((u_k, v_k) \in \mathcal{OD} \ u_k - \text{org} \) is connected to only one departure event and \(v_k - \text{dest} \) is connected to only one arrival event, an optimal solution to TTwR can be found by solving one linear program.

2. For a fixed maximal number of OD-pairs \(m\), TTwR can be solved in polynomial time by solving \(O(|E|^{2m})\) linear programs.

3. In particular, if there is only one OD-pair, an optimal solution to TTwR can be found by solving at most \(|A_{\text{org}}| \cdot |A_{\text{dest}}| = O(|E|^2)\) linear programs.

This means, for real-world applications (TTwR) can be solved efficiently if the planners restrict themselves to the most important OD-pairs. It can also be solved efficiently in regional networks which are sparse in the following sense: Most stops are visited by only one (or very few) lines, and the number of connections in the planning period (e.g. per day) is rather small.

Note that also in the general case, for every OD-pair \((u_k, v_k)\) there might be an exponential number of paths from \(u_k - \text{org}\) to \(v_k - \text{dest}\), but there are at most \(O(|E|^2)\) reasonable virtual activities. These are usually much less in practical applications due to the following reasons: first, origin and destination stations \(u_k\) and \(v_k\) are usually connected only to a relatively small set of departure or arrival events, and second, only such pairs of departure and arrival events have to be considered which can be connected by a path. The latter excludes, e.g. pairs of events where the departure takes place after the arrival.

We can even further reduce the set \(\mathcal{R}_{\text{virt}}\). To this end, for an OD-pair \((u_k, v_k)\) denote by \(P_{u_k}^{\text{upp}}\) a shortest path from \(i_k\) to \(j_k\) in \(\mathcal{N}\), taking upper bounds as activity lengths and let \(l_{u_k}^{\text{upp}}\) denote its length.
Lemma 6.4. Let \((\pi^*, R^*)\) be an optimal solution to an instance of \(TTwR\) and let \(P_k^*\) denote the path for \(OD\)-pair \((u_k, v_k)\) in \(OD\). Then it holds that
\[
c_{TTwR}(\pi^*, P_k^*) \leq l_{upp}^k
\]
for all \((u_k, v_k) \in OD\).

Proof.
\[
c_{TTwR}(\pi^*, P_k^*) \leq c_{TTwR}(\pi^*, P_{upp}^k) = \sum_{(i,j) \in P_{upp}^k} (\pi^*_j - \pi^*_i) \leq \sum_{(i,j) \in P_{upp}^k} u_{(i,j)} = l_{upp}^k.
\]

Since the length of a path is bounded from below by the sum of lower bounds on its activities, for an \(OD\)-pair \((u_k, v_k) \in OD\) it suffices to consider the virtual activities
\[
A_{virt}^k := \{(i,j) : (u_k - org, i) \in E_{org}, (j, v_k - dest) \in E_{dest}, \exists \text{ directed path } P \text{ from } i \text{ to } j \text{ in } N \text{ with } \sum_{a \in P} l_a \leq l_{upp}^k\},
\]
i.e., it suffices to compare the optimal timetables for all virtual routings in
\[
A_{virt}^k := \{(i^1, j^1), (i^2, j^2), \ldots, (i^n, j^n) : (i^k, j^k) \in A_{virt}^k\}.
\]

To enhance the enumeration approach, we can extend the LP formulation given in Corollary 6.2 to include the choice of a virtual activity. To this end, additionally to the timetable variables
\[
\pi_i := \text{point in time when operational event } i \text{ takes place} \quad \forall i \in E_{op}
\]
for every \((u_k, v_k) \in OD\) we introduce variables
\[
z_{ij}^k := \begin{cases} 1 & \text{if virtual activity } (i, j) \text{ is used for OD-pair } (u_k, v_k) \\ 0 & \text{otherwise.} \end{cases}
\]
for all \((i,j) \in A_{virt}^k\) and
\[
t^k := \text{travel time of OD-pair } (u_k, v_k).
\]
The virtual-activity-based IP is given as
\[
\begin{align*}
\min & \quad \sum_{(u,v) \in OD} w_k t^k, \\
\text{subject to} & \quad l_{ij} \leq \pi_j - \pi_i \leq u_{ij} \quad \forall (i,j) \in A, \\
& \quad \sum_{(i,j) \in A_{virt}^k} z_{ij}^k = 1 \quad \forall (u_k, v_k) \in OD, \\
& \quad t^k \geq \pi_j - \pi_i - M^k(1 - z^k_{ij}) \quad \forall (i,j) \in A_{virt}^k, (u_k, v_k) \in OD, \\
& \quad z_{ij}^k \in \{0,1\} \quad \forall (i,j) \in A_{virt}^k, (u_k, v_k) \in OD, \\
& \quad \pi_i, t^k \in \mathbb{Z} \quad \forall i \in E_{op}, (u_k, v_k) \in OD
\end{align*}
\]
Like in formulation (3-5) for \(TTF\), the constraints (9) ensure feasibility of the timetable \(\pi\). Constraints (10) ensure that exactly one virtual activity is chosen for every \(OD\)-pair. Constraints (11)
set the variables \( t^k \) to the travel time of OD-pair \((u_k, v_k)\) if \( M^k \) is chosen big enough for every \((u_k, v_k) \in \mathcal{OD}\), i.e., if

\[
M^k \geq (\pi_{j_1} - \pi_{i_1}) - (\pi_{j_2} - \pi_{i_2})
\]

\( \forall (i_1, j_1), (i_2, j_2) \in A^k_{virt} \) and every feasible timetable \( \pi \) for \( \mathcal{N} \).

(14)

It is easy to see that, e.g., \( M^k := \sum_{a \in A_{op}} u_a \) for all \((u_k, v_k) \in \mathcal{OD}\) is a valid choice for \( M^k \) since

\[
(\pi_{j_1} - \pi_{i_1}) - (\pi_{j_2} - \pi_{i_2}) = \sum_{(i,j) \in P_{i_1,j_1}} (\pi_j - \pi_i) - \sum_{(i,j) \in P_{i_2,j_2}} (\pi_j - \pi_i)
\]

\leq \sum_{(i,j) \in P_{i_1,j_1}} (\pi_j - \pi_i)

\leq \sum_{(i,j) \in P_{i_1,j_1}} u_{ij}

\leq \sum_{a \in A_{op}} u_a

for every feasible timetable \( \pi \).

Furthermore, the minimal values \( M^k_{\text{min}} \) for (14) can be calculated in polynomial time by solving a series of linear programs:

\[
\text{max } M^k((i_1, j_1), (i_2, j_2)) := (\pi_{j_1} - \pi_{i_1}) - (\pi_{j_2} - \pi_{i_2}),
\]

s.t. \( \pi_h - \pi_g \geq l_{gh} \) \( \forall (g,h) \in \mathcal{A} \),
\( \pi_h - \pi_g \leq u_{gh} \) \( \forall (g,h) \in \mathcal{A} \),
\( \pi_g \in \mathbb{R} \) \( \forall g \in \mathcal{E}_{op} \),

for all \((i_1, j_1), (i_2, j_2) \in A^k_{virt} \times A^k_{virt} \) and setting

\[
M^k_{\text{min}} := \max_{((i_1, j_1),(i_2, j_2)) \in A^k_{virt}, \times A^k_{virt}} M^k((i_1, j_1), (i_2, j_2)).
\]

### 7 Numerical results

In this section we evaluate our solution approach. We follow two goals: We show that integrated timetabling and routing can be solved optimally in small computation times for medium-sized real-world problems using the virtual-activity-based formulation (8)-(13) in many cases. Moreover, we compare our approach to a flow-based integer programming formulation which is similar to the formulations used in [6, 20, 39] for delay management with rerouting and periodic timetabling, respectively. Our computations were done using Gurobi 4.5.1 [12] on a dual core CPU AMD Opteron running with 2.2 Ghz with 9.8 GB main memory. In the following, we first describe our test instances, then state the flow-based formulation (see (15)-(23)) and thereafter discuss the sizes and computation times of the two formulations.

Our test instances were generated based on data provided by the LinTim toolbox, which is a collection of test instances and algorithms for public transportation problems [19]. The infrastructure of the first type of instances, named A-x, is based on the metro network in Athens, Greece. These instances differ only in the number of considered OD-pairs. The infrastructure of the second type of instances, B-x is based on the German rail traffic, the instances differ in the considered time period. The line plans and the corresponding input data for TTwR were calculated using LinTim. The size of the considered instances is reported in Table 1.

In order to compare the virtual-activity-based formulation (8)-(13) with other approaches we also implemented the following flow-based formulation which is similar to the formulation of [20, 39]
for periodic timetabling with integrated passenger routing and to the approach in [6] for delay management with rerouting. Also in this formulation, we have variables π, representing the timetable, but passengers are represented by multi-commodity flows in the EAN, i.e., flow variables q_a^k specify whether OD-pair (u_k, v_k) uses activity a or not and variables t_{org}^k and t_{dest}^k specify the departure time at the origin u_k and the arrival time at the destination v_k for every OD-pair (u_k, v_k). The flow-based IP is given as

$$\min \sum_{(u_k, v_k) \in \mathcal{OD}} w_k (t_{\text{dest}}^k - t_{\text{org}}^k),$$

s.t. $l_{ij} \leq \pi_j - \pi_i \leq u_{ij}$ \quad \forall (i, j) \in \mathcal{A},

$$\sum_{a \in \delta^-(u_k^a)} q_a^k = 1 \quad \forall (u_k, v_k) \in \mathcal{OD},$$

$$\sum_{a \in \delta^-(i)} q_a^k - \sum_{a \in \delta^+(i)} q_a^k = 0 \quad \forall i \in \mathcal{E}_{\text{op}}, (u_k, v_k) \in \mathcal{OD},$$

$$\sum_{a \in \delta^+(v_{\text{dest}}^k)} q_a^k = 1 \quad \forall (u_k, v_k) \in \mathcal{OD},$$

$$t_{\text{org}}^k \leq \pi_i + M_{\text{org}}^k (1 - q_a^k (u_k^a)) \quad \forall i \in \mathcal{E}_{\text{org}},$$

$$t_{\text{dest}}^k \geq \pi_i - M_{\text{dest}}^k (1 - q_a^k (i, v_{\text{dest}}^k)) \quad \forall i \in \mathcal{E}_{\text{dest}},$$

$$q_a^k \in \{0, 1\} \quad \forall (u_k, v_k) \in \mathcal{OD}, a \in \mathcal{A},$$

$$\pi_i, t_{\text{org}}^k, t_{\text{dest}}^k \in \mathbb{Z}, \quad \forall i \in \mathcal{E}_{\text{op}}, (u_k, v_k) \in \mathcal{OD}.$$
to the flow-based formulation, where we set \( l_b := \sum_{(u_k, v_k) \in OD} w_k l_k^{\text{flow}} \) for \( l_k^{\text{flow}} \) denoting the length of a shortest path from \( u_k^{\text{op}} \) to \( v_k^{\text{dest}} \) in \( N \) with respect to arc lengths \( L_a := l_a \forall a \in A_{\text{op}} \) and \( L_a := 0 \forall a \in A_{\text{org}} \cup A_{\text{dest}} \).

We now compare the virtual-activity-based formulation with the flow-based formulation. In order to abbreviate notation, we denote \( \mathcal{E}^k := \mathcal{E}_{\text{org}}^k \cup \mathcal{E}_{\text{dest}}^k \) for every \((u_k, v_k) \in OD\) and write \( \sum_k |\mathcal{E}^k| := \sum_{(u_k, v_k) \in OD} |\mathcal{E}^k| \) in both tables.

Table 2 shows the number of variables in the flow-based formulation (15)-(23) and in the virtual-activity-based formulation (8)-(13). The first column shows the variables \( \pi_i \) for \( i \in E \) which define the timetable. The number of these variables is the same for both formulations. It is also the same for all instances A-x, since these instances rely on the same network and differ only in the number of OD-pairs. In the head of the table, we see the number of variables of each type as a function in the input size. The following rows report the values for the instances described in Table 1. Furthermore, the overall number of variables is given.

Table 3 reports the number of constraints in the flow-based formulation (15)-(23) and the virtual-activity-based formulation (8)-(13). In the first column we see the number of timetable feasibility constraints used in both IP formulations. The head of Table 3 shows the number of constraints of the two formulations (15)-(23) and (8)-(13) as a function in the input size. The subsequent rows report the values for the instances described in Table 1. Furthermore, the overall number of constraints is given.

Note that adding flow constraints for every OD-pair causes a significant increase in the number of constraints for the flow-based formulation, while only \(|OD|\) constraints are added for the choice of the virtual activities in the virtual-activity-based formulation. In total, this leads to a higher overall number of constraints in the flow-based formulation, compared to the virtual-activity-based one.

However, in the instances A-x the number of the 'bigM'-constraints (20)-(21) and (11), which set the departure and arrival times \( r_{\text{org}}^k \) and \( r_{\text{dest}}^k \) in the flow-based model and the travel time \( l_k^k \) in the virtual-activity-based model, is about one third smaller in the flow-based formulation than in the virtual-activity-based formulation. In contrast to this result, in the instances B-x the number of 'bigM'-constraints (20)-(21) in the flow-based formulation is much larger compared to the number of 'bigM'-constraints (11), in the virtual-activity-based formulation. Note that in these instances there are more OD-pairs than in the instance A-x, but the number of virtual activities is small in comparison. Since there is at least one virtual activity for every OD-pair, for at least 19 out of 29 OD-pairs in instance B-1 and 134 out of 213 OD-pairs there is only one virtual activity,
Table 4: Optimality gap and time needed to solve the test instances to optimality (in minutes and seconds). ‘-’ signifies that time limit of two hours was reached.

<table>
<thead>
<tr>
<th></th>
<th>flow-based (15)-(23)</th>
<th>virtual-activity-based (8)(13)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A-1</td>
<td>0% 6 m 34 s 13 m 11 s</td>
<td>0% 0 m 08 s 0 m 14 s</td>
</tr>
<tr>
<td>A-2</td>
<td>16.2% - -</td>
<td>0% 0 m 18 s 0 m 26 s</td>
</tr>
<tr>
<td>A-3</td>
<td>0% 13 m 36 s 31 m 42 s</td>
<td>0.2% - -</td>
</tr>
<tr>
<td>A-4</td>
<td>&lt;0.1% - -</td>
<td>0% 7 m 17 s 7 m 42 s</td>
</tr>
<tr>
<td>B-1</td>
<td>0% &lt;0 m 01 s 37 m 14 s</td>
<td>0% &lt;0 m 01 s 0 m 02 s</td>
</tr>
<tr>
<td>B-2</td>
<td>0% 0 m 02 s 1284 m 55 s</td>
<td>0% &lt;0 m 01 s 0 m 06 s</td>
</tr>
</tbody>
</table>

which makes constraints (10) and (11) trivial for these OD-pairs. Hence for these instances we are almost in the situation of TTE.

In Table 4, we see the time needed by Gurobi to solve the described instances of TTwR, using the flow-based formulation (15)-(23) and the virtual-activity-based formulation (8)-(13), and the overall times, including the time needed for building the IP. The entry ‘-’ signifies that within the time limit of 2 hours no optimal solution was found. We also report the optimality gap for the best found solution within two hours. For the calculations we used $M := \sum_{a \in A_{op}} u_a$ for constraints (20)-(21) and (11). The use of smaller values for the ‘bigM’-constraints did not improve the running time (see [1] for details).

We note that for the flow-based formulation, the time to build up the IP is very long, due to the high number of variables and constraints. This observation is extreme for the instances B-x: Once the IP was built, it could be solved within two seconds, but due to the long time needed for building it, the overall solution time was much longer. Since the number of variables and constraints in the virtual-activity-based formulation was much smaller for all test instances, we did not face any such problems there. However, also if we regard only the solution time needed by Gurobi, we observe that except for the instance A-3, the virtual-activity-based formulation could solve the test instances A-x much faster than the flow-based formulation.

For further details and a theoretical and experimental analysis of different bounds on the ‘bigM’-values, see [1, 30].

8 Summary and further research

In this paper we considered the problem of finding a timetable and a passenger routing simultaneously with the objective of minimizing the total travel time. Compared to traditional timetabling approaches where passengers’ routes are assumed to be known, this approach models passengers’ behavior more accurately and leads to better solutions since the passengers’ route choice is integrated in the optimization process.

Even for the aperiodic case, timetabling with routing is shown to be strongly NP-hard and not approximable in polynomial time unless $P = NP$. However, a slight modification of the problem leads to the problem of timetabling with routing between events which is solvable in polynomial time. This result can be exploited in order to solve TTwR exactly by integer programming or by an enumeration method for which polynomially solvable cases are identified.

First computational experience shows that the virtual-activity-based formulation is a promising approach which performs well for real-world test instances. Further results are under investigation.

Another approach to find a solution is to start with some routing, then choose a timetable, then again a routing and so on. Also this approach is under investigation from a theoretical and an experimental point of view, see [1, 30].

Another line of research deals with the routing decisions of the passengers. The assumption that a passenger uses a path with minimal travel time is only a rough approximation and does not
always coincide with the passengers’ behavior observed in practice. The integration of routing
models which reflect the passengers’ behavior more realistically is hence a promising goal.

Acknowledgments

We want to thank the LinTim team and in particular Jennifer Anhalt for providing data and
computing the numerical results.

References


17


<table>
<thead>
<tr>
<th>Number</th>
<th>Authors</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>2012 - 1</td>
<td>R. Hübner, A. Schöbel</td>
<td>When is rounding allowed? A new approach to integer nonlinear optimization</td>
</tr>
<tr>
<td>2012 - 2</td>
<td>M. Goerigk, A. Schöbel</td>
<td>Improving the Modulo Simplex Algorithm for Large-Scale Periodic Timetabling</td>
</tr>
<tr>
<td>2012 - 3</td>
<td>M. Goerigk, M. Schachtebeck, A. Schöbel</td>
<td>Dependencies between line planning, timetabling and delay management: Simulations with the LinTim toolbox</td>
</tr>
<tr>
<td>2012 - 4</td>
<td>M. Goerigk</td>
<td>A Local Search Algorithm for Robust Periodic Timetabling</td>
</tr>
<tr>
<td>2012 - 5</td>
<td>M. Goerigk, S. Westphal</td>
<td>A Combined Local Search and Integer Programming Approach to the Traveling Tournament Problem</td>
</tr>
<tr>
<td>2012 - 6</td>
<td>M. Goerigk, A. Schöbel</td>
<td>A Note on the Relation between Strict Robustness and Adjustable Robustness</td>
</tr>
<tr>
<td>2012 - 7</td>
<td>M. Schmidt, A. Schöbel</td>
<td>The complexity of integrating passenger routing decisions in public transportation models</td>
</tr>
<tr>
<td>Number</td>
<td>Authors</td>
<td>Title</td>
</tr>
<tr>
<td>----------</td>
<td>----------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>2012 - 8</td>
<td>F. Werner, T. Hohage</td>
<td>Convergence rates in expectation for Tikhonov-type regularization of Inverse Problems with Poisson data</td>
</tr>
<tr>
<td>2012 - 9</td>
<td>M. Schmidt,</td>
<td>Line planning with equilibrium routing</td>
</tr>
<tr>
<td>2012 - 10</td>
<td>M. Pazouki, R. Schaback</td>
<td>Bases for Conditionally Positive Definite Kernels</td>
</tr>
<tr>
<td>2012 - 11</td>
<td>B. Zwicknagl, R. Schaback</td>
<td>Interpolation and Approximation in Taylor Spaces</td>
</tr>
<tr>
<td>2012 - 12</td>
<td>K. Böhmer, R. Schaback</td>
<td>A Nonlinear Discretization Theory</td>
</tr>
<tr>
<td>2012 - 13</td>
<td>M. Bozzini, M. Rossini, R. Schaback</td>
<td>Generalized Whittle-Matern and Polyharmonic Kernels</td>
</tr>
<tr>
<td>2012 - 14</td>
<td>D. Mirzaei, R. Schaback</td>
<td>Solving Heat Conduction Problems by the Direct Meshless Local Petrov-Galerkin (DMLPG) method</td>
</tr>
<tr>
<td>2012 - 15</td>
<td>J. Puerto, A. Schöbel, S. Schwarze</td>
<td>On equilibria in generalized Nash games with applications to games on polyhedral sets</td>
</tr>
<tr>
<td>2012 - 16</td>
<td>T. Dollevoet, D. Huisman, A. Schöbel, M. Schmidt</td>
<td>Delay Management including Capacities of Stations</td>
</tr>
<tr>
<td>2012 - 17</td>
<td>A. Schöbel, D. Scholz</td>
<td>A general solution algorithm for non-convex mixed integer optimization problems with only few continuous variables</td>
</tr>
<tr>
<td>Number</td>
<td>Authors</td>
<td>Title</td>
</tr>
<tr>
<td>----------</td>
<td>------------------------</td>
<td>------------------------------------------------------------</td>
</tr>
<tr>
<td>2012 - 18</td>
<td>A. Schöbel, S. Schwarze</td>
<td>Finding Delay-Resistant Line Concepts using a Game-Theoretic Approach</td>
</tr>
<tr>
<td>2012 - 20</td>
<td>M. Schmidt</td>
<td>Simultaneous Optimization of Delay Management Decisions and Passenger Routes</td>
</tr>
<tr>
<td>2012 - 24</td>
<td>M. Schmidt, A. Schöbel</td>
<td>Timetabling with Passenger Routing</td>
</tr>
</tbody>
</table>