Ordering Constraints in Time Expanded Networks for Train Timetabling Problems

Frank Fischer

Algorithmic Algebra and Discrete Mathematics
University of Kassel
Heinrich-Plett-Str. 40, 34132 Kassel, Germany
`frank.fischer@uni-kassel.de`

Abstract

The task of the *train timetabling problem* is to find conflict free schedules for a set of trains with predefined routes in a railway network. This kind of problem has proven to be very challenging and numerous solution approaches have been proposed. One of the most successful approaches is based on time discretized network models. However, one of the major weaknesses of these models is that fractional solutions tend to change the order of trains along some track, which is not allowed for integer solutions, leading to poor relaxations. In this paper, we present an extension for these kind of models, which aims at overcoming these problems. By exploiting a configuration based formulation, we propose to extend the model with additional *ordering constraints*. These constraints enforce compatibility of orderings along a sequence of tracks and greatly improve the quality of the relaxations. We show in some promising preliminary computational experiments that our approach indeed helps to resolve many of the invalid overtaking problems of relaxations for the standard models.

1998 ACM Subject Classification G.1.6 [Numerical Analysis] Optimization

Keywords and phrases combinatorial optimization, train timetabling, Lagrangian relaxation, ordering constraints

Digital Object Identifier 10.4230/OASIcs.ATMOS.2015.97

1 Introduction

Given a railway infrastructure network and a set of trains with fixed routes, the train timetabling problem (TTP) aims at determining schedules for each train such that certain operational restrictions like station capacities and headway times are satisfied, see, e. g., the recent surveys [14] and [5]. The possible goals for these schedules vary. Typical ones are to have the trains arrive as early as possible (often a goal for freight trains) [7] or follow a given ideal timetable with as less delay as possible (passenger trains) [4].

One major approach for large scale instances is based on time expanded networks for modeling train schedules [2, 6, 11]. These models give rise to huge integer programming formulations and cannot be solved directly by standard solvers. Recently, several mathematical techniques, e. g., dynamic graph generation [9], bundle methods [10], rapid branching [1, 17], were developed to overcome this situation.

In this paper we present an extension for time expanded models. One major drawback of these models is that they do not contain variables representing orderings of trains running on the same infrastructure arc or using the same station. This has the consequence that fractional relaxations (e. g., linear relaxations or Lagrangian relaxations) tend to find (fractional) solutions that allow overtaking of trains on tracks where it is not possible for integral solutions. This behavior is expected, but leads to very weak relaxations. Because of the

15th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS'15).
Editors: Giuseppe F. Italiano and Marie Schmidt; pp. 97–110

OpenAccess Series in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

lack of ordering variables it is not easy to strengthen the model with additional constraints that model the combinatorial properties of the infrastructure network, i. e., forbid arbitrary changes of the order of trains running on the same tracks.

Borndörfer and Schlechte [2] proposed a variant of the time expanded models, in which headway constraints on tracks are not enforced by inequality constraints, which forbid succeeding trains running in too quick succession. Instead they use so called *configuration networks*, which model sets of non-conflicting train runs on a single infrastructure arc. This means, while inequality constraints provide an "outer description" of feasible train runs, configurations networks form an "inner description", modeling all feasible points. Although both models are equivalent from a theoretical point of view, the latter has an additional advantage. Configuration networks form some kind of extended formulation and add further variables and constraints to the model. In particular, they allow easy access to a limited set of ordering variables for trains. Indeed, a configuration network provides decision variables that model whether two trains follow each other in direct succession.

In this paper we propose a model that exploits these ordering variables to forbid invalid changes of the ordering of trains along a sequence of succeeding tracks. We show that this model provides a stronger relaxation than previous models, resolving some of the nastiest weaknesses of linear relaxations for this kind of models for the TTP.

This paper is organized as follows. We give a formal description of the problem in Section 2 and state the basic time expanded model in Section 3. In particular, the new model extension with ordering constraints is described in Sections 3.2 and 3.3. Then we sketch our solution approach in Section 4 and present some promising preliminary computational experiments in Section 5. Finally, we conclude our paper in Section 6.

2 Problem Description

We briefly recall a formal description of the TTP next. The *infrastructure network* is a directed graph $G^I = (V^I, A^I)$, where the nodes V^I represent stations, junctions, and crossings and the arcs A^I represent connecting railway tracks. Arcs are directed because the running times of some trains may depend on the direction of travel and because some tracks are single line (one physical track that is used in both directions). In the latter case it is important to distinguish between trains running in the same and in opposite directions. Furthermore, we are given a set of trains R and each train $r \in R$ is associated with a path $G^r = (V^r, A^r) \subseteq G^I$ in the infrastructure network, its route, and a starting time $\hat{t}^r \in T$. Denote by R^a the set of all trains running on $a \in A^I$. Between two trains $r, r' \in R^a$ running on a there is a minimal headway time $h^a(r, r') \in \mathbb{N}$ (in discrete time steps), which is the minimal time between the two trains entering this track (we assume that $h^a(r, r') > 0$ for all $a \in A^I$, $r, r' \in R^a$, and that the triangle inequalities $h^a(r, r') + h^a(r', r'') \geq h^a(r, r'')$ for each three trains $r, r', r'' \in R^a$ are satisfied). As mentioned above, the infrastructure network may contain double line tracks (arcs with one physical track for each direction) and single line tracks (arcs with only one physical track for both directions). In the case of single line tracks we set $h^{(u,v)} = h^{(v,u)}$ and the headway times for trains running in opposite directions are appropriately high, so that no two such trains may occupy the track at the same time. Furthermore, there are capacity constraints on the nodes that state that at most a certain number of trains $c_u \in \mathbb{N}$ may be at the same station $u \in V^I$ at the same time. Note that in contrast to other works the schedules of the trains are completely free. There do not exist already fixed trains or conditions stating that schedules must not deviate too much from some ideal timetable (see, e. g., [4]).

▶ Remark. Note that the train routes are only a rough estimate of the reality. For instance, we do not consider the exact routing of trains through junctions as well as the possibility of small adjustments to the train routes, e. g., by choosing one of several parallel tracks. These aspects are often interesting in practice and can be incorporated in our model, but for sake of simplicity we decided to neglect these details.

In our experiments we consider only a simple but reasonable objective function, which aims at minimizing the delay of each train at each station. Here delay means the difference of the arrival time at some station compared with the earliest possible arrival time, if a train does not wait at some earlier station. The exact definition is given below in Section 3.1.

3 Model

In this section we present our time discretized model for the TTP. We start with the networks modeling the schedules of each train and some basic constraints in Section 3.1. Next we review the modeling of headway constraints based on configuration networks in Section 3.2. In Section 3.3 we present the new ordering constraints and finally the complete model in Section 3.4.

3.1 Basic Time Expanded Model

One of the most successful models in the literature for solving the TTP is based on time expanded networks, see, e. g., [6, 2]. Given a set of discrete time steps $T = \{1, \dots, |T|\}$ (usually minutes), we have for each train $r \in R$ a *time expanded network* $G_T^r = (V_T^r, A_T^r)$ where $V_T^r = V^r \times T$ and

$$\begin{aligned} A_T^r = &\{((u,t_u),(v,t_v)) \colon (u,v) \in A^r, t_u, t_v \in T, t_v - t_u = \bar{t}^r_{(u,v)}\} \\ &\cup \{((u,t_u),(u,t_u+1)) \colon u \in V^r_{\text{wait}}, t_u, t_u+1 \in T, t_u \geq \hat{t}^r\}, \end{aligned}$$

with $V^r_{\text{wait}} \subseteq V^r$ the nodes at which r might wait and $\bar{t}^r_{(u,v)} \in \mathbb{N}_0$ the *running time* of r over track $(u,v) \in A^r$. A feasible schedule of train r then corresponds to a path $P \subseteq G_T^r$ from the first to the last station. In particular, let $\hat{u}^r$, $\check{u}^r$ denote the first and the final station of r, respectively, then the variables $x^r \in \{0,1\}^{A_T^r}$ have to satisfy the following *flow conservation constraints*

$$\sum_{\substack{(v,t_v)\in V_T^r:\\ e=((v,t_v),(u,t_u))\in A_T^r}} x_e^r = \sum_{\substack{(v,t_v)\in V_T^r:\\ e=((u,t_u),(v,t_v))\in A_T^r}} x_e^r, \quad (u,t_u) \in V_T^r \setminus \{(\hat{u}^r,\hat{t}^r)\}, u \neq \check{u}^r, \tag{1a}$$

$$\sum_{\substack{(v,t_v)\in V_T^r:\\ e=((\hat{u}^r,\hat{t}^r),(v,t_v))\in A_T^r}} x_e^r = 1, \tag{1b}$$

$$\sum_{t\in T} \sum_{\substack{(v,t_v)\in V_T^r:\\ e=((v,t_v),(\check{u}^r,t))\in A_T^r}} x_e^r = 1. \tag{1c}$$

Constraints (1a) are the flow conservation constraints on all intermediate nodes. Constraint (1b) states that the path must start at node $(\hat{u}^r, \hat{t}^r) \in V_T^r$ and constraint (1c) enforces that the path must end at some node $(\check{u}^r, t) \in V_T^r$ corresponding to its final station. We denote the set of all feasible paths in G_T^r by

$$\mathcal{P}^r := \{x^r \in \{0,1\}^{A_T^r} \colon x^r \text{ satisfies constraints (1a)–(1c)}\}.$$

We associate a binary variable $x_a^r \in \{0,1\}$ with each $a \in A_T^r$ in each time expanded network, where $x_a^r = 1$ if and only if a is contained in the timetable of train r.

The capacity constraints in the nodes enforce that at each time instance $t \in T$ at most $c_u \in \mathbb{N}$ trains may be in $u \in V^I$ at the same time. Hence, the sum over all arcs representing a train being in u at time t

$$K(u,t) := \{(r,a) \colon a = ((u',t'),(u,t)) \in A_T^r, r \in R\}$$

must be at most c_u

$$\sum_{(r,a)\in K(u,t)} x_a^r \le c_u, \qquad u \in V^I, t \in T. \tag{2}$$

The headway restrictions impose that certain arcs must not be contained in the final timetable simultaneously if they correspond to some train runs violating a headway constraint. In particular, let $(r,a) \in A_T^r$ and $(r',a') \in A_T^{r'}$ be two arcs with $a = ((u,t_u),(v,t_v))$ and $a' = ((u,t'_u),(v,t'_v))$ with $t'_u - t_u < h^{(u,v)}(r,r')$, then those arcs must not be used both. Therefore, we have the following *headway constraints* for each pair of incompatible arcs

$$x_a^r + x_{a'}^{r'} \le 1, \qquad \{(r,a),(r',a')\} \in H, \tag{3}$$

where H is the set of pairs of incompatible train arcs.

The objective for all trains is to run as fast possible in order to minimize all delays. For this we use the simple objective function described in [7]. Let $(u,v) \in A^r$ be a track segment of train $r \in R$ and $\underline{t}_v \in \mathbb{R}_+$ the earliest possible arrival time of r at v (i. e., the arrival time if r starts at $\hat{t}^r$ at its first station and does not wait at any station before v). Then the penalty of a run arc $e = ((u,t_u),(v,t_v)) \in A_T^r$, $u \neq v$, is the quadratic delay weighted with the relative length of the track segment compared with the length of the whole train run. Let $\ell_e^r = \bar{t}_{(u,v)}^r$ denote the length (in terms of running time) of track e and $\ell^r := \sum_{a\in A^r} \bar{t}_a^r$ denote the minimal running time for the whole run. We set the weight of arc $e = ((u,t_u),(v,t_v)) \in A_T^r$, $u \neq v$, to

$$w_e^r := -\ell_e^r/\ell^r \cdot (t_v - \underline{t}_v)^2,$$

and all other weights to 0 (note that we use negative weights because we want to have a maximization problem).

Putting all together, the TTP can be formulated as integer program as follows:

$$\begin{aligned} \text{maximize} \quad & \sum_{r\in R} \langle w^r, x^r \rangle \\ \text{subject to} \quad & x^r \in \mathcal{P}^r, \qquad r \in R, \\ & (2), (3), \end{aligned}$$

i. e., we select for each train r a feasible schedule $x^r \in \mathcal{P}^r$, so that all paths satisfy the headway and capacity constraints.

However, we do not use the headway inequalities on all arcs $a \in A^I$, but we use another approach to be presented in Section 3.2.

3.2 Configuration Networks

Our modeling of headway constraints is based on an extended formulation, which has been proposed in [2]. This formulation introduces additional *configuration networks* in order to model the safety distances between succeeding trains.

The construction is as follows: Let s_a be an artificial source and t_a an artificial sink node on track $a = (u, v) \in A^I$. The set

$$\tilde{A}^a_r := \{((u, t_u), (v, t_v)) \colon ((u, t_u), (v, t_v)) \in A^r_T, (u, v) = a\}$$

denotes all running arcs of train $r \in R$ on track a. These arcs correspond to some arcs of the configuration network. For each arc $e \in \tilde{A}^a_r$, $r \in R^a$, we introduce a pair of start and end nodes

$$B^a := \{(r, \text{start}, e) \colon r \in R^a, e \in \tilde{A}^a_r\}, \quad \text{and,} \quad E^a := \{(r, \text{end}, e) \colon r \in R^a, e \in \tilde{A}^a_r\}.$$

For these nodes we define the following sets of arcs:

1. The *start arcs* $\tilde{A}^a_{\text{start}} := \{(s_a, u) \colon u \in B^a\}$.
2. The *end arcs* $\tilde{A}^a_{\text{end}} := \{(u, t_a) \colon u \in E^a\}$.
3. The *wait arcs*

$$\tilde{A}^a_{\text{wait}} := \{((r, \text{start}, e), (r, \text{start}, e')) \in B^a \times B^a \colon e = ((u, t_u), (v, t_v)) \in \tilde{A}^a_r, e' = ((u, t_u + 1), (v, t_v + 1)) \in \tilde{A}^a_r\},$$

   which allow to have a larger distance between two succeeding trains than the minimal headway time.
4. The run arcs, each corresponding to a possible run of one train

$$\tilde{A}^a_{\text{run}} := \{((r, \text{start}, e), (r, \text{end}, e)) \in B^a \times E^a \colon r \in R^a\}.$$

5. The *headway arcs*

$$\tilde{A}^a_{\text{hw}} := \{((r, \text{end}, e), (r', \text{start}, e')) \in E^a \times B^a \colon r, r' \in R^a, r \neq r', e = ((u, t_u), (v, t_v)), e' = ((u, t'_u), (v, t'_v)), t'_u - t_u = h^a(r, r')\},$$

   which model that r runs immediately before r' while respecting the headway time.

Then the configuration network $\tilde{G}^a = (\tilde{V}^a, \tilde{A}^a)$, $a \in A^I$, is defined by

$$\tilde{V}^a := \{s_a, t_a\} \cup B^a \cup E^a,$$
$$\tilde{A}^a := \tilde{A}^a_{\text{start}} \cup \tilde{A}^a_{\text{end}} \cup \tilde{A}^a_{\text{run}} \cup \tilde{A}^a_{\text{hw}} \cup \tilde{A}^a_{\text{wait}}.$$

Figure 1 shows an example of a configuration network for two trains.

Given that the headway times are strictly positive and transitive, it is easy to see that valid configurations on track $a \in A^I$, i. e. selections of exactly one run for each train on that arc such that the headway restrictions are satisfied, correspond to s_a-t_a-paths that contain exactly one run arc for each train, see [16]. Therefore, we define the set of feasible configuration paths in $\tilde{G}^a$, $a \in A^I$, as

$$\tilde{\mathcal{P}}^a := \{P \subseteq \tilde{G}^a \colon P \text{ is an } s_a\text{-}t_a\text{-path in } \tilde{G}^a \text{ with exactly one run arc for each train}\}.$$

▶ Remark. Note that for single line tracks, only one configuration network is constructed for both arcs $(u, v) \in A^I$ and $(v, u) \in A^I$.

As with the train graphs we associate the binary variables $\tilde{x}^a_e \in \{0, 1\}$ with the arcs $\tilde{A}^a$. The coupling between configuration networks and train graphs is simple: a train may use one of its run arcs if and only if the corresponding run arc is contained in the configuration

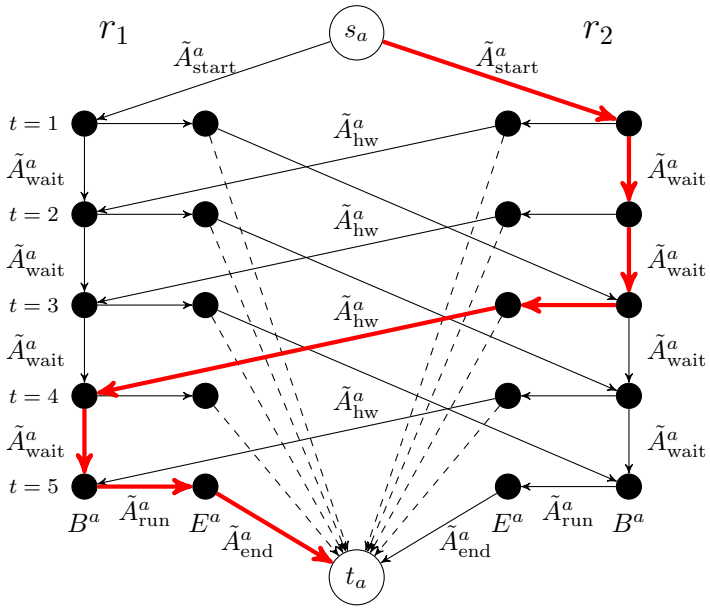

Figure 1 Example configuration network for two trains $\{r_1, r_2\} = R^a$ on some infrastructure arc $a \in A^I$. The headway times are $h^a(r_1, r_2) = 2$ and $h^a(r_2, r_1) = 1$. The red path corresponds to a configuration with r_2 running at $t = 3$ followed by r_1 at $t = 5$.

for this arc. Hence, denoting the run arc in a configuration network for some train arc $e = ((u, t_u), (v, t_v)) \in A^r_T$, $r \in R$, $(u, v) \in A^I$, by

$$\text{cfg}(e) := ((r, \text{start}, e), (r, \text{end}, e)) \in \tilde{A}^{(u,v)},$$

we have the following *configuration constraints*

$$x^r_e = \tilde{x}^{(u,v)}_{\text{cfg}(e)}, \qquad r \in R, e = ((u, t_u), (v, t_v)) \in A^r_T.$$

3.3 Ordering Constraints

The basic observation when using configuration networks is the following. The run of a train $r \in R$ on some specific infrastructure arc $a \in A^I$ is represented by the run arcs $((r, \text{start}, e), (r, \text{end}, e)) \in \tilde{A}^a_{\text{run}}$. However, we are interested in the headway arcs $((r, \text{end}, e), (r', \text{start}, e')) \in \tilde{A}^a_{\text{hw}}$. If one of these arcs equals 1, then this means that train r' is the direct successor of r on arc a. In particular, with

$$\tilde{A}^a_{\text{hw}}(r, r') := \{((r, \text{end}, e), (r', \text{start}, e')) \in \tilde{A}^a_{\text{hw}}\}$$

we define the *ordering variables*

$$s^a_{r,r'} = \sum_{a' \in \tilde{A}^a_{\text{hw}}(r,r')} \tilde{x}^a_{a'}, \qquad a \in A^I, r, r' \in R^a,$$

with the interpretation

$$s^a_{r,r'} := \begin{cases} 1, & r' \text{ succeeds } r \text{ directly on } a \in A^I, \\ 0, & \text{otherwise.} \end{cases}$$

One of the weaknesses of the standard time expanded formulation is that combinatorial properties of the network are not represented well. Figure 2 shows a typical situation: two

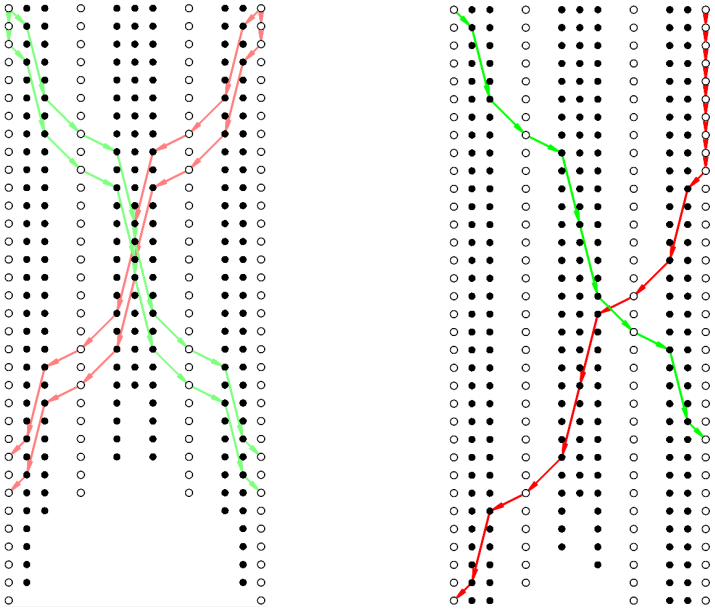

Figure 2 Tiny example with two trains running in opposite directions on a single line track with two passing possibilities. The white nodes have capacity 2 the other nodes have capacity 1. Nodes in the same row correspond to a sequence of stations at the same time step, nodes in the same column correspond to one station but at different time steps (time grows from top to bottom). All arcs between black nodes are single track, the arcs adjacent to white nodes are double track. The left picture shows an optimal solution for the standard linear relaxation. Note that the solution exploits the weak formulation allowing the two trains to meet and pass on a single line part. The right picture shows the optimal solution if ordering constraints are added to all consecutive paths of single line tracks: one train has to wait at a capacity 2 node for the other train to pass.

trains run on a sequence of single line tracks, such that no overtaking is allowed on the intermediate nodes. For instance, the intermediate nodes could be small local stations without overtaking/passing possibility or the arcs represent a sequence of blocking areas (guarded by signals) which must not be occupied by more than one train. In particular, stopping and waiting at these intermediate nodes is allowed. Obviously, because there is no overtaking possibility, in a feasible solution one train must go first through the complete sequence of tracks and then the other. However, the fractional solution can easily exploit the weak formulation: both trains can run fractionally in short succession and pass at an intermediate node, see Figure 2.

Ordering variables provide an easy way to formulate these kind of non-overtaking properties. Consider a path of nodes $(u_1, u_2, \dots, u_n)$ such that the capacity of each intermediate node is 1, i. e. $c_{u_i} = 1$, $i = 2, \dots, n-1$, and all arcs (u_i, u_{i+1}), $i = 1, \dots, n-1$, are single line tracks. Then it is clear that the order of trains running on arc (u_1, u_2) (or (u_2, u_1), which is the same because it is a single line track) must be equal to the order of trains running on arc (u_{n-1}, u_n). This can be enforced using the following *ordering constraints*

$$s_{r,r'}^{(u_1,u_2)} = s_{r,r'}^{(u_{n-1},u_n)}, \qquad r, r' \in R^{(u_1,u_2)} = R^{(u_{n-1},u_n)}.$$

Note that exactly the same constraints give rise to ordering conditions if the path consists of double line tracks. The only difference is that the configuration networks associated with (u_1, u_2) and (u_{n-1}, u_n) only model the headway conditions (and thus the ordering) of trains

running in the same direction. Thus there would be two sets of ordering constraints, one for (u_1, u_2) and (u_{n-1}, u_n), and one for (u_2, u_1) and (u_n, u_{n-1}).

▶ Remark. Of course, the ordering of the trains must not only be the same on the first and last arcs of the path but also on each intermediate arc, so one would have ordering constraints for all pairs of arcs $\{(u_i, u_{i+1}), (u_j, u_{j+1})\}$, $i, j = 1, \dots, n-1$, $i \neq j$. However, because of practical considerations (configuration networks can get large and are thus expensive from a computational point of view), we use them only on the first and last arcs of a path.

In our preliminary experiments we used only this simplest case of ordering constraints on paths, where no overtaking/passing is possible. This may sound oversimplified, but it is a typical situations in real world instances as well, where connections between different stations are made of such paths. However, it should be possible to extend this approach to situations, where the capacity of some intermediate station u_i, $i \in \{2, \dots, n-1\}$, is a small number larger than 1. For instance, if $c_{u_i} = 2$ on exactly one intermediate station, then the orders of trains on the first and last arcs may differ, but not arbitrarily. In particular, in this case the order of three trains may not be reversed (r_1 before r_2 before r_3 on the first arc but r_3 before r_2 before r_1 on the last arc). However, in order to express this kind of ordering restrictions one might need more complex configuration subproblems that do not only have variables for trains in direct succession but also for trains with an additional train in between. Solving such configuration problems (or a reasonable approximation of them) will probably be hard in itself and needs more work.

3.4 Complete Model

In this section we present the complete model. Let $\mathcal{O} \subseteq \binom{A^I}{2}$ be the set of pairs of infrastructure arcs, such that for $\{(u, u'), (v, v')\} \in \mathcal{O}$ there is a path $P = (u_1, \dots, u_n)$ of maximal length with $c_{u_i} = 1$ for all $i = 2, \dots, n-1$ such that

1. $(u, u') = (u_1, u_2)$ and $(v, v') = (u_{n-1}, u_n)$,
2. either all arcs are single line tracks or all arcs are double line tracks,
3. $R^{(u,u')} = R^{(u_i, u_{i+1})}$ for all $i = 1, \dots, n-1$.

Let $\tilde{A}^I := \bigcup_{X \in \mathcal{O}} X$ be the set of all arcs that are contained in at least one ordering constraint. Because configuration networks enlarge the model quite a bit, we use them only on those infrastructure arcs, where they are required to formulate some ordering constraints. On all other arcs $A^I \setminus \tilde{A}^I$ we use classical headway inequalities. The resulting model reads

$$
\begin{aligned}
\text{(TTP)} \qquad \text{maximize} \quad & \sum_{r \in R} \langle w^r, x^r \rangle, && && (4)\\
\text{subject to} \quad & x^r \in \mathcal{P}^r, && r \in R, && (5)\\
& \tilde{x}^a \in \tilde{\mathcal{P}}^a, && a \in \tilde{A}^I, && (6)\\
& x^r_a + x^{r'}_{a'} \leq 1, && a \in A^I \setminus \tilde{A}^I, \{(r,a),(r',a')\} \in H, && (7)\\
& x^r_e = \tilde{x}^{(u,v)}_{\mathrm{cfg}(e)}, && e = ((u,t_u),(v,t_v)) \in A^r, r \in R, && (8)\\
& \sum_{(r,a) \in K(u,t)} x^r_a \leq c_u, && u \in V^I, t \in T, && (9)\\
& s^a_{r,r'} = \sum_{a' \in \tilde{A}^a_{\mathrm{hw}}(r,r')} \tilde{x}^a_{a'}, && a \in \tilde{A}^I, r, r' \in R^a, && (10)\\
& s^a_{r,r'} = s^{a'}_{r,r'}, && r, r' \in R^a = R^{a'}, \{a, a'\} \in \mathcal{O}. && (11)
\end{aligned}
$$

We optimize a linear objective function (4) so that in each train graph and each configuration network a feasible path representing a schedule (5) or a configuration (6), respectively, is selected. The configuration networks and train graphs are coupled by (8), for infrastructure arcs $a \in \tilde{A}^I$ that have a configuration network. On the other arcs we use the usual headway inequalities (7). The capacity restrictions on the nodes are enforced by (9). Finally, constraints (10) introduce the ordering variables which are then coupled by the ordering constraints (11). Note that we write the constraints (10) only for the sake of presentation as they can easily be substituted in (11).

We use the objective function of [7], which is quite simple: all trains should run as fast as possible, so that any delays are minimized. In other words, the weights are so that early run arcs have higher weights than later arcs.

4 Solution Methods

In this section we briefly describe our solution method. The basic approach is to apply Lagrangian relaxation to (TTP), see, e. g., [3] for an early work using this approach. Indeed, we relax all coupling constraints (7)–(9) and (11) (with (10) being substituted in (11)). We collect all coupling equality and inequality constraints in

$$\sum_{r\in R} M_{1,r}x^r + \sum_{a\in\tilde{A}^I} M_{1,a}\tilde{x}^a = b_1 \qquad \text{and} \qquad \sum_{r\in R} M_{2,r}x^r + \sum_{a\in\tilde{A}^I} M_{2,a}\tilde{x}^a \le b_2,$$

respectively. The dual problem then reads

$$\begin{aligned} \text{(LR)} \qquad & \text{minimize} && \varphi(y,z), \\ & \text{subject to} && y \in \mathbb{R}^{m_1}, z \in \mathbb{R}^{m_2}_+, \end{aligned}$$

where the dual function $\varphi(y,z)$ is defined by

$$\varphi(y,z) := \sum_{r\in R} \max_{x^r\in\mathcal{P}^r} \langle w^r - M_{1,r}^T y - M_{2,r}^T z, x^r\rangle + \sum_{a\in\tilde{A}^I} \max_{\tilde{x}^a\in\tilde{\mathcal{P}}^a} \langle -M_{1,a}^T y - M_{2,a}^T z, \tilde{x}^a\rangle.$$

It is well-known that the dual problem (LR) is a non-smooth, convex optimization problem that can be solved by, e. g., bundle methods [13]. In particular, we use a special scaling variant of a bundle method based on ConicBundle [12]. Because there is a huge number of potential coupling constraints, they are separated during the solution process.

In order to solve (LR) one has to evaluate the function φ at certain trail points (y,z) provided by the bundle method. The subproblems in the train graphs G^r, $r \in R$, have the form

$$\max_{x^r\in\mathcal{P}^r} \langle w^r - M_{1,r}^T y - M_{2,r}^T z, x^r\rangle,$$

which are longest path problems in acyclic networks (which are equivalent to shortest path problems with the negated objective function because of the acyclicity). Because these networks get very large if the number of time steps increases, we use a dynamic graph generation algorithm proposed in [9], implemented in the DynG callable library [8].

The subproblems in the configuration networks $\tilde{G}^a$, $a \in \tilde{A}^I$, read similarly

$$\max_{\tilde{x}^a\in\tilde{\mathcal{P}}^a} \langle -M_{1,a}^T y - M_{2,a}^T z, \tilde{x}^a\rangle.$$

However, the set of feasible paths $\tilde{\mathcal{P}}^a$, $a \in \tilde{A}^I$, is more complicated in general than the sets $\mathcal{P}^r$, $r \in R$. The reason is that a feasible configuration corresponds to an s_a-t_a-path if and

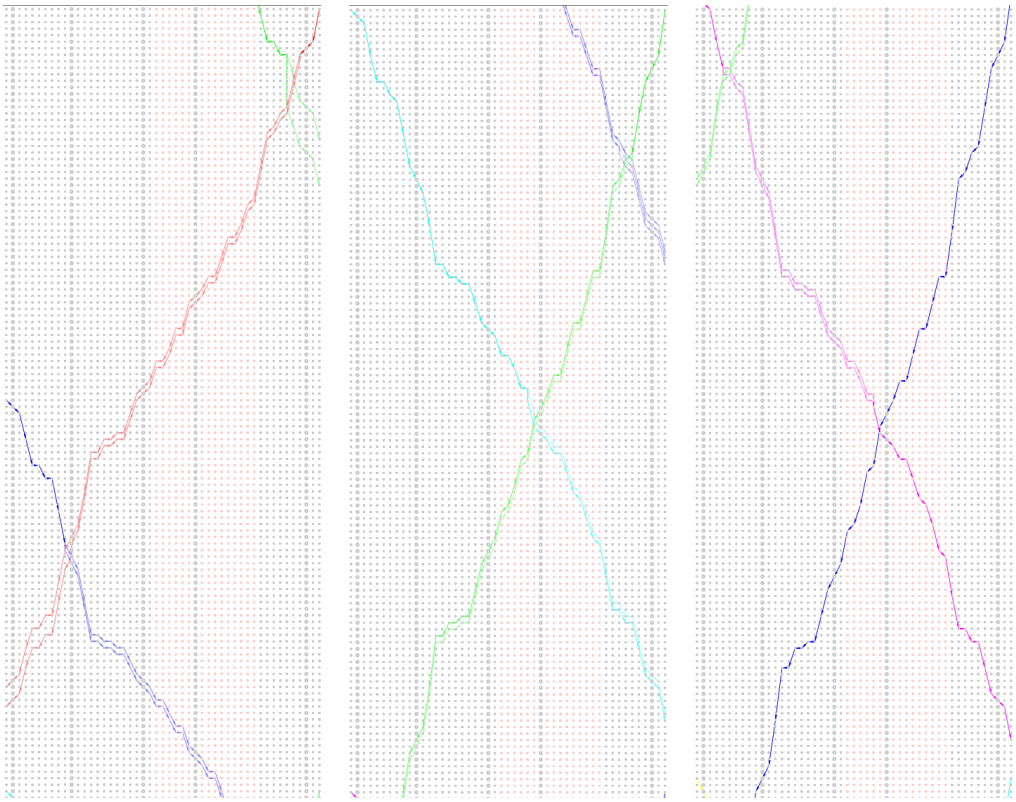

Figure 3 Part of the solution of (LR) for 12 trains *without ordering constraints* (consecutive part from left to right picture). The gray nodes form single line parts, the red nodes are a double line part. Only the thicker nodes have capacity 2, all others have capacity 1. Several passings on the single line parts are not resolved correctly because of the weak relaxation.

only if this path contains exactly one run arc for each train. These subproblems become very difficult to be solved exactly even for relatively small numbers of trains, say ten. Therefore we use the following relaxed version. Let

$$\tilde{\mathcal{P}}^a_{\text{rlx}} := \left\{ P \subseteq \tilde{G}^a : P \text{ is an } s_a\text{-}t_a \text{ containing at most } |R^a| \text{ run arcs of } \tilde{A}^a_{\text{run}} \right\} \supseteq \tilde{\mathcal{P}}^a.$$

This problem is simpler (it is a rather easy constrained shortest path problem), and we solve it using a dynamic programming algorithm, exploiting the fact that $\tilde{G}^a$ is acyclic. However, solving these subproblems only approximately can lead to worse solutions of the relaxation in practice, see Section 5.

5 Numerical Tests

We tested our approach on a small 12 train test instance from the RAS Problem Solving Competition 2012 [15]. The subproblems for the train graphs have been implemented with the DynG callable library [8], the subproblems in the configuration networks are solved approximately by a dynamic programming approach (see Section 3.4). The Lagrangian relaxation (LR) has been solved using a proximal bundle method based on ConicBundle [12]. All experiments are done on an Intel Core i7 @ 3.5 GHz with 12 GB RAM.

The test instance consists of a corridor with 49 nodes and arcs and has several single line parts and one double line part. There are only 4 passing points in the single line parts and

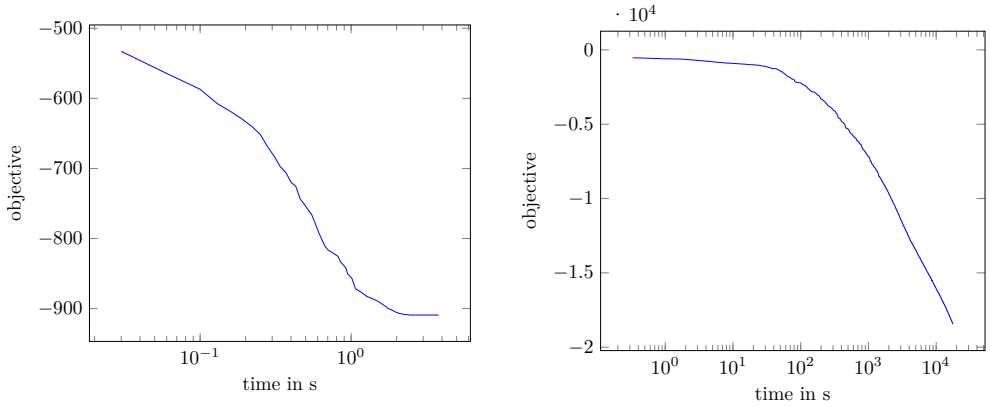

Figure 4 The left picture shows the dual function value after a certain amount of time in seconds for the model without ordering constraints. The right picture shows the same for the model with ordering constraints. Note that the time is given in a logarithmic scale. The model without ordering constraints converges very quickly, but the optimal value is quite bad. The model with ordering constraints converges much slower, but the objective value is much better. In fact, the dual bounds are better than without ordering constraints already after a few seconds.

one overtaking point in the double line part. The trains run over a period of about 9 hours, and each train requires between 1 and 2 hours to go from one end of the network to the other, depending on its speed. The model uses a time discretization of one minute.

First we look at the quality of the models in terms of the dual bounds. Figure 4 shows the development of the dual bounds after some computation time.

The pictures show that the model without ordering constraints converges very quickly to an optimal solution. In contrast, the model with ordering constraints converges very slowly but generates much better bounds. In fact, the bounds are better than the bounds without ordering constraints after 10 seconds. The slow convergence is a known disadvantage of configuration based models (also see [7], Chapter 6.4.3) when used in a Lagrangian relaxation approach with first-order optimization methods (like bundle methods).

However, in order to investigate the structure of the optimal solutions, we looked at the approximate solutions after a large computation time. A part of the resulting schedule is shown in Figure 3 for the model *without ordering constraints* and in Figure 5 for the model *with ordering constraints*.

The pictures show that without the ordering constraints the (fractional) solution of the relaxation easily exploits the weakness of the model and lets the trains meet and pass in the middle of single line parts hardly slowing down any train. In contrast, the model with the ordering constraints successfully finds appropriate waiting possibilities for some of the trains, so that meet and pass points are exactly at the nodes with capacity 2 or within the double line part.

However, the results are not always perfect. Figure 6 shows another part of the solution. Here the meet and pass decisions have not been resolved correctly. But the reason for this is not the inaccuracy of the ordering constraints. The problem is that the configuration subproblems are only solved approximately. When looking at the solutions of the subproblems during the algorithm, one sees that there are paths in $\tilde{G}^a$, $a \in \tilde{A}^I$, which do not correspond to correct configurations. In particular, the returned paths contain more than one run arc $((r, \text{start}, e), (r, \text{end}, e)) \in \tilde{A}^a_{\text{run}}$ for some train $r \in R^a$ and zero run arcs $((r', \text{start}, e'), (r', \text{end}, e')) \in \tilde{A}^a_{\text{run}}$ for some other train $r' \in R^a$ (see Section 4).

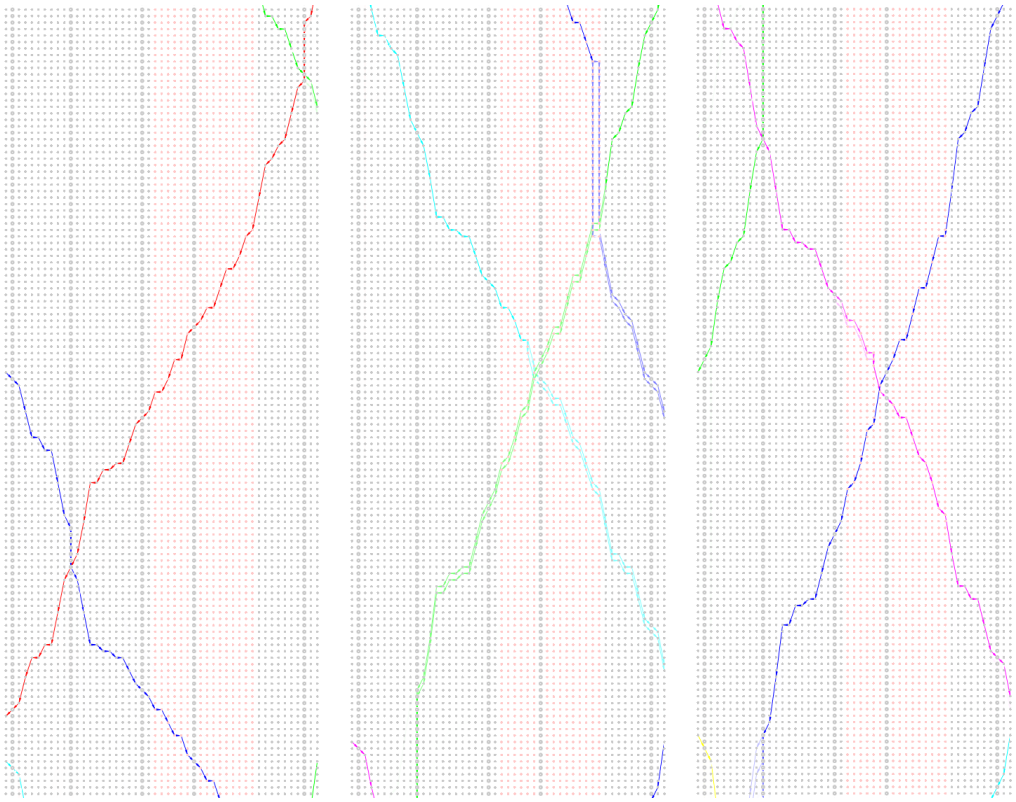

Figure 5 Part of the solution of (LR) for 12 trains *with ordering constraints* (consecutive part from left to right picture). The gray nodes form single line parts, the red nodes are a double line part. Only the thicker nodes have capacity 2, all others have capacity 1. The passings of the trains have been resolved correctly.

Note that we were not able to solve the relaxation with solving the configuration subproblems exactly. Although solving one subproblem exactly takes only few seconds, the solution method using a proximal bundle method requires many iterations and thus many subproblem evaluations for all arcs in $\tilde{A}^I$, hence the solution process did not have sufficient progress in reasonable time.

▶ Remark. Indeed, it turned out that the solution of the configuration subproblems, even if we solve them only approximately, were the main bottleneck in our approach. In contrast, the solution of the train graph subproblems was extremely fast thanks to the used dynamic graph generation technique, which ensures that only very small parts of the train graphs have to be stored and that the solutions can be found very quickly.

6 Conclusions and Future Work

In this paper we proposed an extension of a configuration network based formulation for the TTP. In particular, we use the configuration networks to formulate constraints that forbid changes in the order of trains along a path of tracks that are not possible due to the existing overtaking possibilities of the infrastructure network. We implemented the model and demonstrated how the new model greatly improves the solution of the Lagrangian relaxation of the model. Indeed, in the example the relaxation resolves many meet and pass

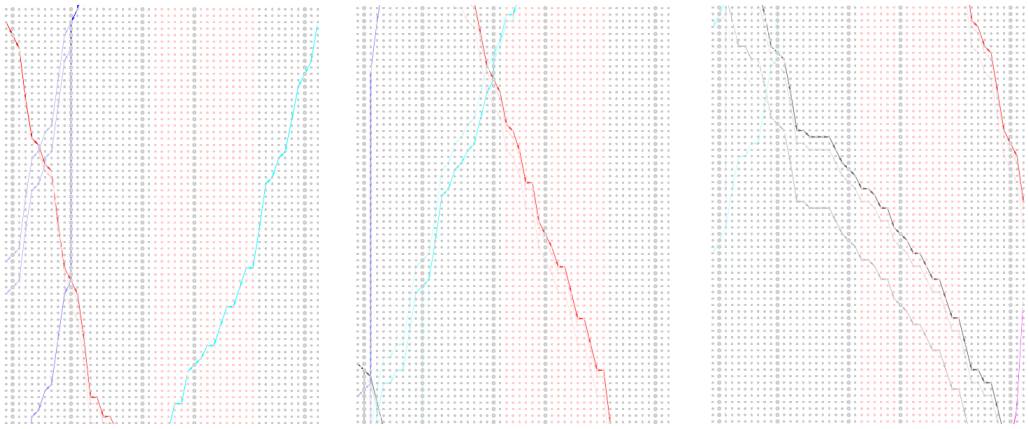

Figure 6 Another part of the solution of (LR) with ordering constraints. Because the configuration subproblems are only solved approximately, the resulting fractional solution may still contain wrong passing decisions (in particular for the blue train in this example).

decisions correctly, which does not happen without the ordering constraints. However, we have also seen that the ordering constraints might not be sufficient to ensure correct meet and pass decisions if the configuration subproblems are not solved exactly.

We can conclude from our experiments that the proposed extension with ordering constraints has great potential to improve existing models. From a theoretical point of view, several improvements of our model are possible. For instance, one could formulate configuration subproblems that represent not only the ordering of trains in direct succession but also of larger groups of trains. This would allow to formulate ordering conditions if a certain sequence of infrastructure arcs has not zero but a small number of overtaking points. Another possible improvement could be to solve the configuration subproblems exactly for small subsets of all trains running on some infrastructure arc. For instance, one could formulate configuration subproblems that ensure that each subset of 3 to 5 trains runs correctly. Given that at most points in time only few trains compete for some infrastructure arc, the exact solution of these small subproblems could be possible from a computational point of view and improve the quality of the relaxation.

Finally, our experiments shows that the current approach using standard first-order methods to solve these problems has a very bad convergence behavior. Hence, although better bounds than before can be reached after short computation times, near optimal solutions still take very long. Therefore several algorithmic developments or alternative solution methods are required in order to improve the efficiency of our approach so that it can be applied to large real world instances.

References

1 Ralf Borndörfer, Andreas Löbel, Markus Reuther, Thomas Schlechte, and Steffen Weider. Rapid branching. *Public Transport*, 5(1-2):1–21, 2013. doi:10.1007/s12469-013-0066-8.

2 Ralf Borndörfer and Thomas Schlechte. Models for railway track allocation. In Christian Liebchen, Ravindra K. Ahuja, and Juan A. Mesa, editors, *ATMOS 2007 - 7th Workshop on Algorithmic Approaches for Transportation Modeling, Optimization, and Systems*, Dagstuhl Seminar Proceedings. Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany, 2007. doi:10.4230/OASIcs.ATMOS.2007.1170.

3 U. Brännlund, P. O. Lindberg, A. Nõu, and J. E. Nilsson. Railway timetabling using Lagrangian relaxation. *Transportation Science*, 32(4):358–369, 1998.

4 Valentina Cacchiani, Fabio Furini, and Martin Philip Kidd. Approaches to a real-world train timetabling problem in a railway node. *Omega*, 58:97–110, 2016. doi:10.1016/j.omega.2015.04.006.

5 Valentina Cacchiani and Paolo Toth. Nominal and robust train timetabling problems. *European Journal of Operational Research*, 219(3):727–737, 2012. doi:10.1016/j.ejor.2011.11.003.

6 Alberto Caprara, Matteo Fischetti, and Paolo Toth. Modeling and solving the train timetabling problem. *Operations Research*, 50(5):851–861, 2002.

7 Frank Fischer. *Dynamic Graph Generation and an Asynchronous Parallel Bundle Method Motivated by Train Timetabling*. PhD thesis, Chemnitz University of Technology, 2013. URL: http://nbn-resolving.de/urn:nbn:de:bsz:ch1-qucosa-118358.

8 Frank Fischer. *DynG Dynamic Graph Generation library*, 2014. URL: http://www.mathematik.uni-kassel.de/~fifr/fossils/dyng.

9 Frank Fischer and Christoph Helmberg. Dynamic graph generation for the shortest path problem in time expanded networks. *Mathematical Programming A*, 143(1-2):257–297, 2014. doi:10.1007/s10107-012-0610-3.

10 Frank Fischer and Christoph Helmberg. A parallel bundle framework for asynchronous subspace optimization of nonsmooth convex functions. *SIAM Journal on Optimization*, 24(2):795–822, 2014. doi:10.1137/120865987.

11 Frank Fischer, Christoph Helmberg, Jürgen Janßen, and Boris Krostitz. Towards solving very large scale train timetabling problems by Lagrangian relaxation. In Matteo Fischetti and Peter Widmayer, editors, *ATMOS 2008 - 8th Workshop on Algorithmic Approaches for Transportation Modeling, Optimization, and Systems*, Dagstuhl, Germany, 2008. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany. doi:10.4230/OASIcs.ATMOS.2008.1585.

12 Christoph Helmberg. *ConicBundle 0.3.11*. Fakultät für Mathematik, Technische Universität Chemnitz, 2012. URL: http://www.tu-chemnitz.de/~helmberg/ConicBundle.

13 Jean-Baptiste Hiriart-Urruty and Claude Lemaréchal. *Convex Analysis and Minimization Algorithms I & II*, volume 305, 306 of *Grundlehren der mathematischen Wissenschaften*. Springer, Berlin, Heidelberg, 1993.

14 Richard M. Lusby, Jesper Larsen, Matthias Ehrgott, and David Ryan. Railway track allocation: models and methods. *OR Spectrum*, 33(4):843–883, oct 2011. doi:10.1007/s00291-009-0189-0.

15 RAS Problem Solving Competition 2012, 2012. URL: https://www.informs.org/Community/RAS/Problem-Solving-Competition/2012-RAS-Problem-Solving-Competition.

16 Thomas Schlechte. *Railway Track Allocation: Models and Algorithms*. PhD thesis, TU Berlin, 2012.

17 Steffen Weider. *Integration of Vehicle and Duty Scheduling in Public Transport*. PhD thesis, TU Berlin, 2007. URL: http://opus.kobv.de/tuberlin/volltexte/2007/1624/.