Towards Solving Very Large Scale Train Timetabling Problems by Lagrangian Relaxation*

Frank Fischer[1], Christoph Helmberg[1],
Jürgen Janßen[2], Boris Krostitz[2]

[1] Technical University of Chemnitz, Department of Mathematics, 09107 Chemnitz
[2] Deutsche Bahn AG, Konzernentwicklung, GSU 1, 60326 Frankfurt, Stephensonstr. 1

Abstract. The train timetabling problem considered is to find conflict free routes for a set of trains in a given railway network so that certain time window conditions are satisfied. We deal with the very large scale problem of constructing such timetables for the German railway network. A number of restrictions on different train types like freight trains or passenger trains have to be observed, e.g., sequence dependent headway times, station capacities, and stopping times. In order to handle the enormous number of variables and constraints we employ Lagrangian relaxation of the conflict constraints combined with a cutting plane approach. The model is solved by a bundle method; its primal aggregate is used for separation and as starting point for rounding heuristics. We present some promising results towards handling a test instance comprising ten percent of the entire network.

1 Introduction

One of the main tasks in strategic railway planning is timetable construction, i.e., to find feasible arrival and departure times for a set of trains with predefined routes. The generated timetables should satisfy a number of different constraints like headway times and station capacities, passenger train stops should lie in given time windows.

This problem is known in the literature as *Train Timetabling Problem* (TTP) and has received considerable attention in the last decades. The TTP is related to the so called *Periodic Event Scheduling Problem* introduced in [1], where periodic timetables are considered, e.g., for subway or fast-train networks, see [2] for a detailed survey on this topic.

Most approaches to the (non-periodic) TTP are based on formulations in the form of *Integer Linear Programs* (ILP) representing train routes by time discretised networks, see [3, 4, 5, 6]. This helps to deal with headway restrictions. Some authors have shown how other types of constraints like station capacities or prescribed timetables can be handled, see, e.g., [7]. The solution methods include

* This work was supported by the *Bundesministerium für Bildung und Forschung* under grant 03HEPAG4. Responsibility for the content rests with the authors.

ATMOS 2008
8th Workshop on Algorithmic Approaches for Transportation Modeling, Optimization, and Systems
http://drops.dagstuhl.de/opus/volltexte/2008/1585

heuristic and exact branch-and-bound based approaches using LP relaxations and Lagrangian relaxations of the ILP, see [4, 8].

In this paper we deal with very large scale real world instances of the German railway company Deutsche Bahn AG (DB) as they arise in the strategic long term planning process of DB. The largest test instance comprises roughly ten percent of the entire German network with approximately 3000 trains to be scheduled for a time period of six hours.

For instances of this size the (in general exponential) number of constraints ensuring sufficient headway time between successive trains on each track necessitates the use of a primal cutting plane approach. A column generation approach for generating the single train schedules would therefore have to include dynamically the effect of the constraints separated so far, e.g., by solving a shortest path problem on a time discretised network showing the effect of the current constraints on the new variables in each time period. In essence, this is exactly what one obtains by classical Lagrangian relaxation of the conflict constraints, which decomposes the problem into shortest path problems on time discretised networks for each train. In this setting, optimizing the dual Lagrange multipliers by simple subgradient methods is not an option, as we need good approximations to the primal solutions for separating the headway constraints in the primal cutting plane approach. The less classical bundle cutting plane approach of [9] offers exactly what we need: it is a bundle method for optimizing the Lagrange multipliers of the dual and generates at the same time a primal approximate solution, the *primal aggregate*, which we use for primal separation of the headway constraints and station capacities. In our instances we deal with different train types and predefined timetable conditions for some of the trains. The Lagrangian relaxation of the model is solved using the ConicBundle package [10].

Our paper is structured as follows. In the next section we give a formal description of the TTP and introduce our model. Section 3 describes the solution methods and finally in section 4 we present preliminary computational results of our approach on our real world test instances.

2 The Train Timetabling Problem

Our TTP can be formally described as follows. We are given an *infrastructure digraph* $D = (V, A)$ representing the railway network, where V is a finite set of nodes (e.g. stations, track switches, ...) and A is the set of directed arcs representing a direction in which the corresponding track can be used. If the locations corresponding to two nodes $u, v \in V$ are connected by two tracks, one for each direction, both arcs uv and vu are in A. We also have two arcs if they are connected by a single track that is used in both directions, i.e., if these two arcs belong to the same physical track. All arcs of the latter kind are collected in the set $A_S \subseteq A$. Let T be the set of trains and $m(j) \in M$ be the *train-type* of $j \in T$ where $M = M_p \dot{\cup} M_f$ is the union of *passenger train types* M_p and *freight train types* M_f. The train-type classifies the speed, length and other properties of the train as needed for strategic planning. A train may *stop* and possibly wait at a node $u \in V$ or may *pass* through the node without stopping. Let $B_S = \{wait, pass\}$ denote the *stopping behaviours* of a train. For each arc the train may stop or pass at the start or at the end node and we collect these *acceleration modes* in the set $B_R = B_S \times B_S$. Then for each arc $uv \in A$ we are given a mapping $t^R_{uv}\colon M \times B_R \to \mathbb{Z}_+$, where $t^R_{uv}(m, b)$ denotes the *running time* of a train of type m over the arc uv with respect to its acceleration mode b in minutes, and a mapping $t^H_{uv}\colon M \times B_R \times M \times B_R \to \mathbb{Z}_+$, where $t^H_{uv}(m_1, b_1, m_2, b_2)$ denotes the *minimal headway time* of a train of type m_1 with acceleration mode b_1 followed by a train of type m_2 with acceleration mode b_2 in minutes. If $uv \in A_S$, we have additional headway times for two trains passing the arc in opposite directions $t^{HS}_{uv}\colon M \times B_R \times M \times B_R \to \mathbb{Z}_+$ with the same interpretation as above.

For each train $j \in T$ the predefined route is given by the *ordered sequence of nodes* $U(j) = (u^j_1, \ldots, u^j_{n_j})$ with $n_j \in \mathbb{N}$, no other nodes are visited. The timetable for a train may be restricted in the following way. For each node u^j_i we have a *stopping-interval* $I^j_i = [t^{S,j}_i, t^{E,j}_i] \subseteq \mathbb{Z} \cup \{\pm\infty\}$ and a *minimal stopping time* $d^j_i \in \mathbb{Z}_+$. The train j has to arrive at node u^j_i before the end of its stopping interval $t^{E,j}_i$, is not allowed to leave the node before $t^{S,j}_i + d^j_i$ and has to wait at the node for at least d^j_i minutes. A waiting time $d^j_i = 0$ signals that train j does not need to stop at node u^j_i. For freight trains there are no stopping restrictions on the nodes except for the first node u^j_1. Here the interval has the form $I^j_1 = [t^{S,j}_1, \infty]$ specifying the train's starting time.

Important constraints on the timetables arise from the capacity of stations. We denote the *absolute capacity* of a node $v \in V$ by $c_v \in \mathbb{N}$, it specifies the maximal number of trains allowed to visit node v at the same time. In many stations the capacity also depends on the direction from which the trains enter the node. For an arc $uv \in A$ the *directional capacity* of the node v is $c_{uv} \in \mathbb{N}$ and describes the maximal number of trains that may stop at or pass through node v at the same time when arriving over arc uv. Clearly, $c_{uv} \leq c_v$ for all $uv \in A$.

We model the problem in a classical way via time discretised networks for the single train routes and by using coupling constraints for the capacity and headway restrictions. Let $S = \{1, \ldots, N\}$ denote the discretised time steps corresponding to minutes. For each train $j \in T$ we have a network $G^j = (V^j, A^j)$ defined as follows. The node set V^j is a subset of $\{\sigma^j, \tau^j\} \cup (B_S \times \{1, \ldots, n_j\} \times S)$, where σ^j is an *artificial source node* and τ^j an *artificial terminal node*, while, e.g., a node $(wait, i, t)^j$ has to be interpreted as train j stops in node u^j_i at time t.

The arc set A^j is built of the following subsets:

- a set of *waiting arcs* $((wait, i, t)^j, (wait, i, t + 1)^j)$, for each $i \in \{1, \ldots, n_j\}$ and $t \in \{1, \ldots, N - 1\}$ for nodes where the train may stop;
- a set of *running arcs* $\big((b_1, i, t)^j, (b_2, i + 1, t + t^R)^j)\big)$ connecting two successive nodes for $i \in \{1, \ldots, n_j - 1\}$, where $t^R = t^R_{u^j_i, u^j_{i+1}}(m(j), (b_1, b_2))$ gives the running time with respect to the stopping behaviours b_1 and b_2;

- a set of *starting arcs* of the form $(\sigma^j, (wait, 1, t)^j)$ corresponding to feasible starting times at the first node;
- a set of *ending arcs* of the form $((wait, n_j, t)^j, \tau^j)$ collecting all possible arrivals at the last node;
- a set of *infeasible arcs* $((b, i, t)^j, \tau^j)$ allowing the train to go from each intermediate node directly to the terminal node.

Of course, the graph G^j contains only those arcs that are valid for the train j with respect to the stopping intervals and minimal stopping times.

Now we introduce for each arc $a \in \mathcal{A} = \bigcup_{j \in T} A^j$ a binary variable $x_a \in \{0, 1\}$ equal to one if and only if the path associated with train j contains arc a. Let $\delta^+(v)$ and $\delta^-(v)$ denote the (possibly empty) sets of arcs leaving and entering node $v \in \mathcal{V} = \bigcup_{j \in T} V^j$. Likewise we define for $v \in V, t \in S$

$$\delta^-(v, t) = \left\{ ((b', i', t')^j, (b, i, t)^j) \in \mathcal{A} \colon u_i^j = v \right\}$$

which is the set of all train arcs arriving at the infrastructure node v at time t, and for $uv \in A, t \in S$

$$\delta^-(uv, t) = \left\{ ((b', i', t')^j, (b, i, t)^j) \in \mathcal{A} \colon u_{i-1}^j u_i^j = uv \right\}$$

which is the set of train arcs arriving at the infrastructure node v at time t over the arc uv. With appropriate arc costs $w_a, a \in \mathcal{A}$ (see Section 4), the ILP formulation reads (later we prefer the dual to be a minimization problem, so we use maximization here)

$$\text{maximize} \sum_{a \in \mathcal{A}} x_a w_a \tag{1}$$

subject to

$$\sum_{a \in \delta^+(\sigma^j)} x_a = 1, \qquad j \in T, \tag{2}$$

$$\sum_{a \in \delta^+(v)} x_a = \sum_{a \in \delta^-(v)} x_a, \qquad j \in T, v \in V^j \setminus \{\sigma^j, \tau^j\}, \tag{3}$$

$$\sum_{a \in \delta^-(v,t)} x_a \le c_v, \qquad v \in V, t \in S, \tag{4}$$

$$\sum_{a \in \delta^-(uv,t)} x_a \le c_{uv}, \qquad uv \in A, t \in S, \tag{5}$$

$$\sum_{a \in C} x_a \le 1, \qquad C \in \mathcal{C}, \tag{6}$$

$$x_a \in \{0, 1\}, \qquad a \in \mathcal{A}. \tag{7}$$

The set $\mathcal{C}$ contains the (in general exponentially large) family of maximal sets $C \subseteq \mathcal{A}$ that hold pairwise conflicting arcs. We say two arcs

$$((b_1, i_1, t_1)^j, (b_2, i_2, t_2)^j) \in A^j \quad \text{and} \quad ((b_1', i_1', t_1')^{j'}, (b_2', i_2', t_2')^{j'}) \in A^{j'}$$

with $t_1 \leq t'_1$ conflict if either

- $u^j_{i_1} u^j_{i_2} = u^{j'}_{i'_1} u^{j'}_{i'_2} = uv \in A$ and $t_1 + t^H_{uv}(m(j), (b_1, b_2), m(j'), (b'_1, b'_2)) > t'_1$, or
- $u^j_{i_1} u^j_{i_2} = u^{j'}_{i'_2} u^{j'}_{i'_1} = uv \in A_S$ and $t_1 + t^{HS}_{uv}(m(j), (b_1, b_2), m(j'), (b'_1, b'_2)) > t'_1$,

i.e., they violate the headway times.

In the objective function (1) the infeasible arcs should have costs with a sufficiently penalizing effect. Constraints (2) ensure that exactly one path per train will be used. Constraints (3) are the flow conservation constraints. The node capacities are imposed by (4) for the absolute capacities and by (5) for the directional capacities. Finally the *clique constraints* (6) forbid the use of conflicting arcs.

3 Solution Methods

Our solution method is based on the *Lagrangian dual* of the model (1)-(7) obtained by relaxing the coupling constraints (4)-(6). In order to explain the decomposition approach, we collect the coupling constraints (4)-(6) in the system $Dx \leq d$ and denote by $D^j, j \in T$, the columns corresponding to the $x_a, a \in A^j$. Furthermore for $j \in T$

$$\mathcal{X}^j := \left\{ x \in \mathbb{R}^{A^j} : x \text{ fulfills (2), (3) and (7) for fixed } j \right\}$$

represents the set of all feasible flows in Graph G^j. The Lagrangian dual problem reads

$$\min_{y \geq 0} \varphi(y)$$

where

$$\varphi(y) := d^T y + \sum_{j \in T} \varphi_j(y),$$

with

$$\varphi_j(y) := \max_{x \in \mathcal{X}^j} \sum_{a \in A^j} x_a w_a - y^T D^j x. \tag{8}$$

Obviously, the φ_j are convex functions because they are maxima over affine functions. For each y the evaluation of $\varphi(y)$ requires the solution of $|T|$ independent shortest path problems (8). Let $x(y)$ be the optimal solution of the shortest path problems for given y, then $g(y) = d - Dx(y)$ is a subgradient of φ at y.

The ConicBundle library [10] implements a bundle method to solve problems of type

$$\min_{y \geq 0} f(y)$$

where $f(y)$ is a convex function given by a first-order oracle, i.e., for given y the oracle returns $f(y)$ and a subgradient $g(y)$ of f at the point y.

The method generates a sequence $(x_k)_{k\in\mathbb{N}}$ of *primal aggregates* that are convex combinations of the primal optimizers giving rise to the subgradients of the φ_j. For an appropriate subsequence $L \subseteq \mathbb{N}$ each cluster point of $(x_l)_{l\in L}$ lies in the set of optimal solutions of the LP relaxation (if such solutions exist), see [9, 11] for technical aspects. Note, the x_k are in general not feasible for our primal problem because they violate the coupling constraints, but they yield successively better approximations to primal optimal solutions.

Since the number of constraints (6) is exponential, we separate the constraints based on the primal aggregates x_k generated by the bundle method, i.e., we add constraints to the model that we find violated by x_k and that are not yet present in the relaxation, see [9, 12] for more information on separation and convergence aspects in bundle methods.

The capacity constraints (4)-(5) are separated by complete enumeration.

The separation of the maximal clique constraints (6) is not trivial. This is because the headway times t_{uv}^{H} and t_{uv}^{HS} may be different for each train-type and for each stopping behaviour. [13] gives an extensive analysis of the structure of clique constraints arising from headway times in TPP and proves that the time window of interest is bounded by twice the maximum headway time. In our case this may be quite large. Therefore, we use a greedy heuristic to find large violated cliques as described in Algorithm 1. For any arc $a \in \mathcal{A}$ with positive flow value we find all arcs in conflict with a and sort them non-increasingly with respect to their flow value. Starting with the single clique $\{a\}$ we successively try to add the next arc in the sequence to all existing cliques. If the new arc does not enlarge the clique, we add the largest subclique containing it. If an upper limit N_C on the number of cliques is exceeded, we eliminate the cliques of minimal weight. The maximal cliques of each a are added as cutting planes if their weight is greater than one and if they are not yet contained in the problem. The routine is called after each descent step of the bundle method.

The last step is the computation of an integral solution. Let x be a primal aggregate returned by the bundle method. In our current approach we create an ordering of the incoming and outgoing trains for each node based on x as follows. Let $u \in V$ be an arbitrary node and let $j \in T$ be a train with $u = u_i^j$ for some $i \in \{2, \dots, n_j\}$. The *average arrival time* of j at u is given by

$$t_u^{j,-} := \frac{\displaystyle\sum_{\substack{a=((b',i-1,t'),(b,i,t))\in A^j \\ u=u_i^j}} t \cdot x_a}{\displaystyle\sum_{\substack{a=((b',i-1,t'),(b,i,t))\in A^j \\ u=u_i^j}} x_a}.$$

These average arrival times define an *arrival order* on the arriving trains visiting node u, i.e., we say for two trains $j, j' \in T$ visiting node u

$$j \preceq_u^- j' \iff t_u^{j,-} \leq t_u^{j',-}.$$

Algorithm 1 Separation of clique constraints on primal aggregate x

$\mathcal{C}_{all} \leftarrow \emptyset$
for $a \in \mathcal{A}\colon x_a > 0$ **do**
  $A_C \leftarrow \{b \in \mathcal{A}\colon a$ is in conflict with $b\}$
  $\mathcal{C} \leftarrow \{\{a\}\}$ *"$\mathcal{C}$ is the set of new cliques."*
  while $A_C \neq \emptyset$ **do**
    Find $c \in \text{Argmax}\{x_b\colon b \in A_C\}$ *"Find conflicting arc c so that x_c maximal."*
    for $C \in \mathcal{C}$ **do**
      "Try to add c to each clique C."
      if $C \cup \{c\}$ is a clique **then**
        $\mathcal{C} \leftarrow (\mathcal{C} \setminus \{C\}) \cup \{C \cup \{c\}\}$
      else
        "If not possible, find (weight-)maximal subclique of C containing c."
        $\bar{C} \leftarrow \{d \in C\colon d$ conflicts with $c\}$
        $\mathcal{C} \leftarrow \mathcal{C} \cup \{\bar{C}\}$
        if $|\mathcal{C}| > N_C$ **then**
          "Only keep a bounded number of cliques."
          $\mathcal{C} \leftarrow \mathcal{C} \setminus \{\text{argmin}\{\sum_{e \in \bar{C}} x_e\colon \bar{C} \in \mathcal{C}\}\}$
        end if
      end if
    end for
  end while
  $\mathcal{C}_{all} \leftarrow \mathcal{C}_{all} \cup \mathcal{C}$
end for
return $\{C \in \mathcal{C}_{all}\colon \sum_{e \in C} x_e > 1\}$

Similarly one can define *average departure times* and a *departure order* $\preceq_u^+$ on the departing trains. Then we run a simulation on the trains, letting the trains arrive and leave as early as possible with respect to the orderings, headway times, stopping intervals and stopping times. Furthermore the data of the running times is a tight upper bound on the fastest possible traversal of the trains, so in the simulation trains may go slower over an arc. In our first experiments those constraints were never violated, but we observed some unexpected delays of passenger trains, so the approach certainly needs further improvement.

The rounding heuristic is called several times to generate timetables for a group of trains. In particular, first only long-distance trains are simulated and the corresponding arcs in the train graphs are fixed to 1. Then a new relaxation is computed for the non-fixed trains and the rounding heuristic is used to generate a timetable for the short-distance trains. Finally in a third iteration the freight trains are handled in the same way.

Since the rounding heuristic had yielded bad results for some test instance, we tried another simple heuristic. We fix successively those arcs to 1, whose values in the relaxation are above 0.8 or, if no such arc exists, the arc with the largest value. When 95% of the arcs have been fixed, we call the rounding heuristic above to generate an integral solution.

4 Numerical Results

We implemented our model in C++ using Cplex 9.1 [14] and the ConicBundle library [10]. All computations were done on an Intel Xeon Dual Core, 3 GHz, 16 GB RAM. The test data is the south-west subnet of the network of DB (roughly Baden-Wuerttemberg). This subnet has about 10% of the size of the whole German network. We considered three test cases of different size:

1. A small part of the network containing the five most frequently used arcs.
2. The main long-distance and freight traffic route along the river Rhine.
3. The whole subnet.

All tests searched for a timetable of six hours. Table 1 shows the instance sizes (the columns Nodes and Arcs refer to the infrastructure network) as well as the number of variables in the resulting model.

Table 1. Test instances.

Instance	Nodes	Arcs	Passenger	Freight	Variables
1	104	193	242	9	317336
2	656	1210	50	67	2448842
3	2103	4681	2501	659	8990060

The cost coefficients $w_a, a \in \mathcal{A}$, have been chosen so that all trains profit from travelling as early as possible, but our tests were focused on the construction of *feasible* and not necessarily *optimal* timetables.

$$w_a = \begin{cases} -(10000 - i) & a = ((b,i,t)^j, \tau^j) \in \mathcal{A}, i \in \{1, \dots, n_j - 1\} \text{ (infeasible arc),} \\ -(n_j - i) & a = ((b,i,t)^j, (b', i+1, t')^j) \in A^j, i \in \{1, \dots, n_j - 1\} \\ 0 & \text{otherwise.} \end{cases}$$

Because of the large amount of memory that the model requires, we were only able to solve instances 1 and 2 with Cplex. All instances could be solved by the bundle method mentioned above. Table 2 shows the time and the memory required to solve the models.

Table 2. Solution times of the relaxation.

Instance	Cplex	ConicBundle	Size
1	33s	12s	160 MB
2	1945s	341s	1 GB
3	–	2512s	6 GB

In order to illustrate the development and progress of the bundle cutting plane approach we present the development of the objective function in Figure

1, the development of the number of constraints and their violation in Figure 2 and Figure 3.

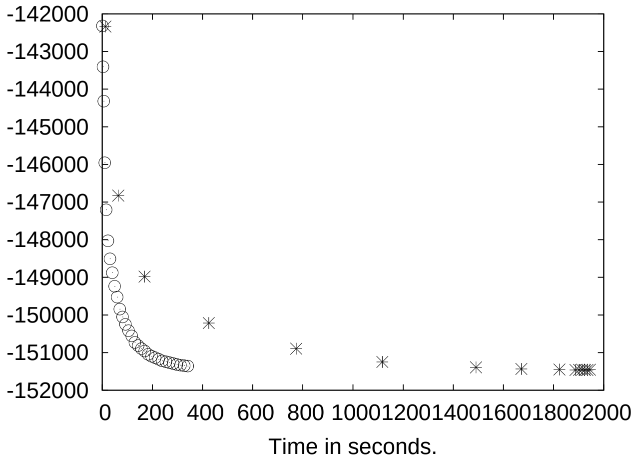

Fig. 1. Development of the objective function for Cplex ($*$) and ConicBundle ($\circ$) of instance 2.

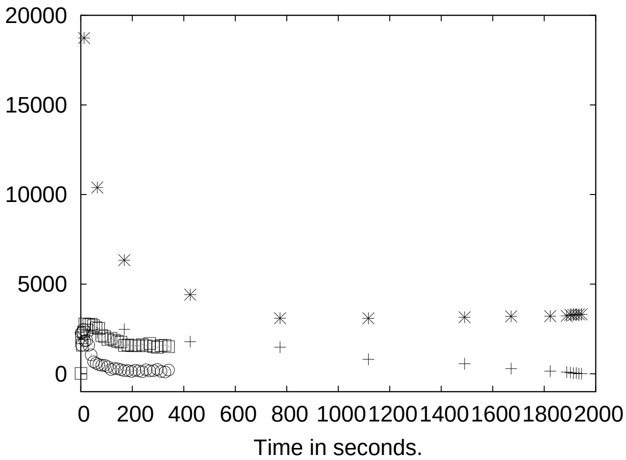

Fig. 2. Development of the number of newly separated (+) and active ($*$) constraints for Cplex and the number of newly separated ($\circ$) and active ($\square$) constraints for ConicBundle of instance 2.

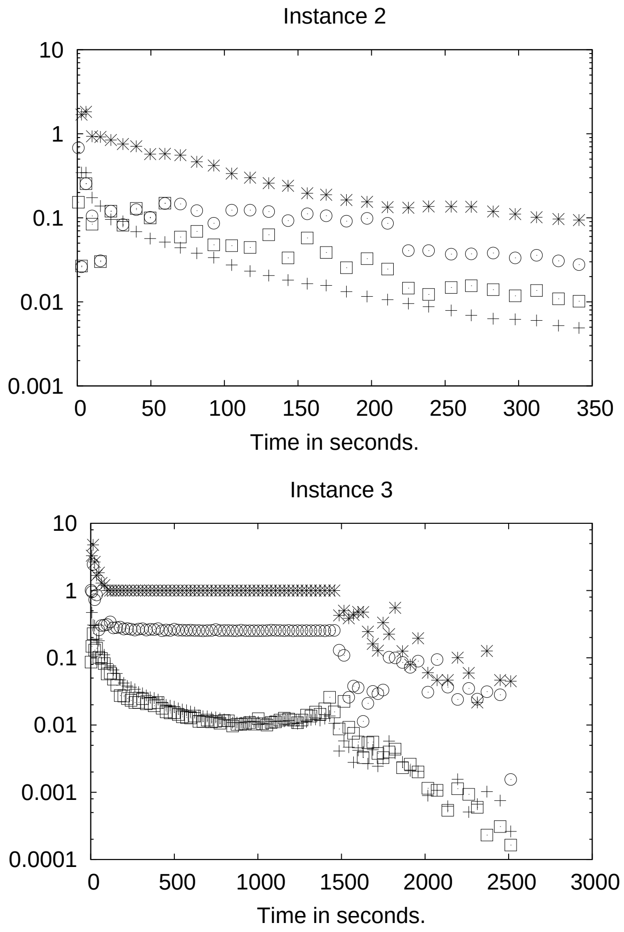

Fig. 3. Development of the maximum (∘) and average (□) violation of capacity constraints and the maximum (∗) and average (+) violation of clique constraints for ConicBundle of instance 2 and 3.

As expected, the objective value shows a strong tailing-off effect which results from the combination of the respective effects for bundle and cutting plane methods. In contrast to the simplex method, the violation of active constraints that are already present in the model stays relatively high over a long time but finally tends to zero in accordance with theory.

Using the rounding heuristic described above, we generated integer solutions for all instances. For instances 1 and 2 the resulting timetable seems to be rather good with almost no delays for the passenger trains (compared with the predefined timetables). Unfortunately, the results for instance 3 are quite bad. In this case, many trains are infeasible (i.e. they use an infeasible arc) and many passenger trains have significant delays. Therefore we used the successive fixing heuristic on that instance. This approach yielded better but not really good results (see table 3). In view of the many further possibilities to exploit structural properties and dual sensitivity information, we are confident that we shall be able to improve in the near future.

Table 3. Results of the rounding heuristic (instances 1, 2, 3) and successive fixing heuristic (instance 3b).

Instance	Time	Infeasible trains	Late trains	Average delays
1	39s	0	0	0
2	697s	0	0	0
3	3182s	40	906	865s
3b	10h	9	778	603s

Remark:

- *Infeasible trains* are those who use infeasible arcs.
- *Late trains* are passenger trains arriving more than 5 minutes later compared with the predefined timetable at at least one station.
- *Average delays* shows the average number of seconds those trains arrive later at their stations compared with the predefined timetable.

5 Conclusion

Relaxations of real world timetabling problems seem to lead to very large scale instances that are not easily solvable by commercial state-of-the-art software, but can be successfully attacked by Lagrangian relaxation combined with bundle methods. For the whole network of DB, however, more work has to be done to reduce the model size on the one hand and to separate the clique constraints more efficiently on the other hand. The solutions of the relaxation show that almost all trains use arcs only in a small time interval, so it should be possible to omit large parts of the train networks at the beginning and generate new arcs dynamically when they are required. More work needs to be invested into developing better rounding heuristics.

Bibliography

[1] Serafini, P., Ukovich, W.: A mathematical for periodic scheduling problems. SIAM J. Discret. Math. **2** (1989) 550–581

[2] Liebchen, C.: Periodic Timetable Optimization in Public Transport. PhD thesis, Technical University Berlin (2006)

[3] Caprara, A., Fischetti, M., Toth, P.: Modeling and solving the train timetabling problem. Oper. Res. **50** (2002) 851–861

[4] Brännlund, U., Lindberg, P.O., Nou, A., Nilsson, J.E.: Railway timetabling using lagrangian relaxation. Transportation Science **32** (1998) 358–369

[5] Caprara, A., Fischetti, M., Guida, P., Monaci, M., Sacco, G., Toth, P.: Solution of real-world train timetabling problems. In: HICSS '01: Proceedings of the 34th Annual Hawaii International Conference on System Sciences (HICSS-34)-Volume 3, Washington, DC, USA, IEEE Computer Society (2001) 3030

[6] Borndörfer, R., Schlechte, T.: Models for railway track allocation. In Liebchen, C., Ahuja, R.K., Mesa, J.A., eds.: ATMOS 2007 - 7th Workshop on Algorithmic Approaches for Transportation Modeling, Optimization, and Systems, Dagstuhl, Germany, Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany (2007)

[7] Cacchiani, V., Caprara, A., Toth, P.: A column generation approach to train timetabling on a corridor. 4OR: A Quarterly Journal of Operations Research **6** (2008) 125–142

[8] Caprara, A., Monaci, M., Toth, P., Guida, P.L.: A lagrangian heuristic algorithm for a real-world train timetabling problem. Discrete Appl. Math. **154** (2006) 738–753

[9] Helmberg, C.: A cutting plane algorithm for large scale semidefinite relaxations. In Grötschel, M., ed.: The Sharpest Cut. MPS-SIAM Series on Optimization. SIAM/MPS (2004) 233–256

[10] Helmberg, C.: ConicBundle 0.2d. Fakultät für Mathematik, Technische Universität Chemnitz. (2008) `http://www.tu-chemnitz.de/~helmberg/ConicBundle`.

[11] Feltenmark, S., Kiwiel, K.C.: Dual applications of proximal bundle methods, including Lagrangian relaxation of nonconvex problems. SIAM J. Optim. **10** (2000) 697–721

[12] Belloni, A., Sagastizábal, C.: Dynamic bundle methods. Mathematical Programming (2008)

[13] Lukac, S.G.: Holes, antiholes and maximal cliques in a railway model for a single track. ZIB-Report ZR 04-18, Zuse-Institut Berlin, Takustraße 7, 14195 Berlin, Germany (2004)

[14] ILOG CPLEX Division 889 Alder Avenue, Suite 200, Incline Village, NV 89451, USA: Using the CPLEX Callable Library. (1997) Information available at `http://www.cplex.com`.