Synthesis of railway infrastructure

J. Spönemann & E. Wendler
Institute of Transport Science, RWTH Aachen University, Germany

Abstract

This paper addresses the problem of generating a cost-optimal railway infrastructure by stating and solving a linear optimization problem. Railway infrastructure is represented by a network consisting of nodes and arcs. The nodes represent stations; the arcs lines connecting the stations. An input instance of the network design problem for railway infrastructure consists of two parts. The stations, which have to be connected in a certain way, and a traffic demand, which relates each pair of nodes (A, B) to a number of trains of different types, has to be routed from A to B in a given time horizon. A newly designed network answers two questions: what is the topology of the network, i.e. which stations are connected to each other and how does the line look like in each connection (e.g. single track, double track, single track with one overtaking station etc.)? The observed kind of routing problem can be stated and solved as a multi-commodity flow problem. In order to get the design of the network using a routing routine, a complete network is constructed. Finding a routing in such a complete network is then equal to designing the network, since the routing chooses the arcs needed and so designs the desired network. To solve the problem efficiently it is stated as a mixed integer program (MIP), which is solvable by standard MIP-solvers.
Keywords: railway infrastructure, strategic long-term planning, network design, multi-commodity flow, MIP.

1 Motivation

A solution of the problem of synthesizing railway infrastructure (SRI) answers the question: what does a cost-optimal network of railway infrastructure for a given traffic requirement look like? Planning a complete new network of infrastructure from scratch is one obvious reason why research in this field pays off. Another one is the strategic long-term planning of infrastructure done by railway infrastructure

WIT Transactions on The Built Environment, Vol 114, © 2010 WIT Press
www.witpress.com, ISSN 1743-3509 (on-line)
doi:10.2495/CR100371

managers. During this planning process, estimated future traffic flows are routed on an existing railway infrastructure to identify bottlenecks or capacity surpluses. After that the infrastructure has to be redesigned to meet the future requirements. As stated by Ross [1], currently long-term infrastructure planning is mainly a *creative* process. The results of this paper create a basis for methods that can provide provable optimal decisions for such planning processes.

The following section, which is the main section of the paper, describes what has to be done to state the SRI problem as a MIP. The third section, which is followed by some conclusions, discusses MIP solving in general and the first results of the solving process.

2 Model

The network of railway infrastructure that has to be designed is very intuitively represented by a graph $G = (V, A)$, with nodes N representing the stations and arcs A representing the lines connecting the stations. Before the model of the problem is presented in detail, let us first have a look at the demands that the input and output of the problem are placing on the structure of the model. An input instance of the SRI problem contains the following information:

- a set of railway stations defined by their distances to each other,
- a set of train types, which are distinguished by parameters such as maximum speed, length, acceleration and deceleration rates,
- a traffic demand consisting of pairs of stations and an associated number of trains – of possibly different types – which should run between these two stations,
- a quality parameter, which limits the degree of utilization for each station-to-station connection,
- a set of stages of extension of lines, which are constructible between two stations, defined by their life cycle cost and capacity, and
- a time horizon.

The questions a solution for this problem has to answer are

- Which stations have to be connected directly to each other?
- What does the connection of two stations look like (e.g. single track, double track, single track with one overtaking station etc.)?
- What does the routing of traffic demand look like, i.e. which route is used by which train to reach its destination station?

It is important to distinguish the two parts of the problems' structure that are mentioned above. On the one hand there is the network design problem, which determines the topology of the network, and on the other hand there is the routing problem of the traffic flow. Let us first consider how the routing problem for traffic flow can be modeled and after that how the network design problem can be solved.

2.1 The routing problem

It is easy to see that finding the best route for a given demand of traffic flow is equal to searching for the minimum cost flow in a network.

2.1.1 The minimum cost flow problem

In the minimum cost flow problem the objective is to find the (s, t)-path with the least cost shipment for a given flow demand, where s and t are the source of the respective target nodes. Ahuja et al. [2] define the minimum cost flow problem as follows:

$$\text{Minimize} \sum_{(i,j)\in A} c_{ij} x_{ij} \tag{1}$$

subject to

$$\sum_{j:(i,j)\in A} x_{ij} - \sum_{j:(j,i)\in A} x_{ji} = b(i) \qquad \forall i \in N, \tag{2}$$

$$l_{ij} \leqslant x_{ij} \leqslant u_{ij} \qquad \forall (i,j) \in A, \tag{3}$$

$$\sum_{i=1}^{n} b(i) = 0, \tag{4}$$

where N is the set of nodes, A the set of arcs, x_{ij} the flow, c_{ij} the cost per unit flow and l_{ij}, u_{ij} the capacity bounds on an arc $(i, j) \in A$. The $b(i)$ in eqn (2) is defined in the following way:

$$b(i) \in \mathbb{N} \begin{cases} < 0, & \text{node } i \text{ is demand node with demand } -b(i), \\ = 0, & \text{node } i \text{ is transshipment node,} \\ > 0, & \text{node } i \text{ is supply node with demand } b(i). \end{cases}$$

The SRI deals with multiple commodities of traffic flow. Each commodity is defined by a start and destination station, as well as by an amount of trains. This leads to a special kind of network flow: the multi-commodity flow.

2.1.2 The multi-commodity flow problem

Ahuja et al. [2] state the multi-commodity flow problem as an optimization problem of the form:

$$\text{Minimize} \sum_{1\leqslant k\leqslant K} c^k x^k \tag{5}$$

subject to

$$\sum_{1\leqslant k\leqslant K} x_{ij}^k \leqslant u_{ij} \qquad \forall (i,j) \in A, \tag{6}$$

$$\mathcal{N} x^k = b^k \qquad k = 1, 2, ..., K, \tag{7}$$

$$0 \leqslant x_{ij}^k \leqslant u_{ij}^k \qquad \forall (i,j) \in A \text{ and } k = 1, 2, ..., K, \tag{8}$$

where $G = (V, A)$ is the network graph, K is the number of commodities, x_{ij}^k the flow of commodity $k \in K$ on arc (i, j), x^k denotes the flow vector of commodity $k \in K$, c^k the corresponding cost per unit flow vector and $\mathcal{N}$ the node-arc incidence matrix, which is used in eqn (7) analogous to eqn (2) to define whether a node is a demand, supply or transshipment node. Eqn (6) restricts the sum of all flows on each arc (i, j) by the upper bound u_{ij}. The values of u_{ij}^k also enable the possibility to bound the flow of each commodity on each arc separately.

2.2 The network design problem

The key idea to solve the network design part of the SRI is to use the solution of the embedded routing problem. To do so a multi-commodity flow problem is solved, getting as input a *complete* network with multiple arcs. As mentioned in the beginning of section 2, the solution of the embedded network design problem of the SRI answers not only the question of which stations have to be connected to each other, but also what a connection looks like. Thereby, different stages of extension for one line, such as *single track*, *double track*, *single track with one overtaking station* and so on are distinguished (see Figure 1).

2.2.1 The multi-arc network

The multi-arc network used for the SRI contains one arc for each stage of extension, which is constructible between two stations. Each of these arcs possesses a different capacity and cost depending on the lines' design and the corresponding life cycle cost. Before the arc capacity is defined in the next subsection, an example is given that shows the working method of solving the network design problem by solving the multi-commodity flow problem on a complete graph with multi-arcs.

2.2.2 Example

Given the graph $G = (\mathcal{V}, \mathcal{A})$, see Figure 2(a), with the set of nodes $\mathcal{V} = \{A, B, C, D\}$ and the set of arcs $\mathcal{A} = \{AB_0, AB_1, AB_2, BC_1, ...\}$, where each arc XY_i of a connection XY has got a different capacity and different cost. Also given three traffic flows C_0, C_1 and C_2 (commodities) with a start and a destination station and an amount of trains (demand). Different train types are indicated by an index. $C_0 = ((A, D), [5_0, 0_1, 0_2])$, $C_1 = ((B, C), [0_0, 15_1, 2_2])$ and $C_2 = ((C, D), [0_0, 0_1, 3_2])$. A routing found by solving a multi-commodity flow problem could, for example, route the flows 5_0, $5_0 + 15_1 + 2_2$ and $5_0 + 3_2$ via arcs

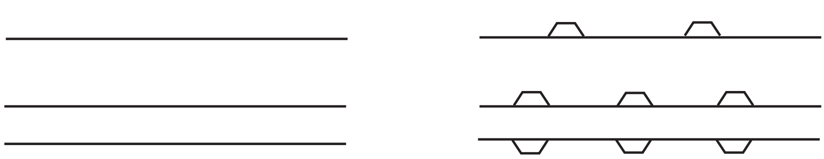

Figure 1: Different stages of extension for one line.

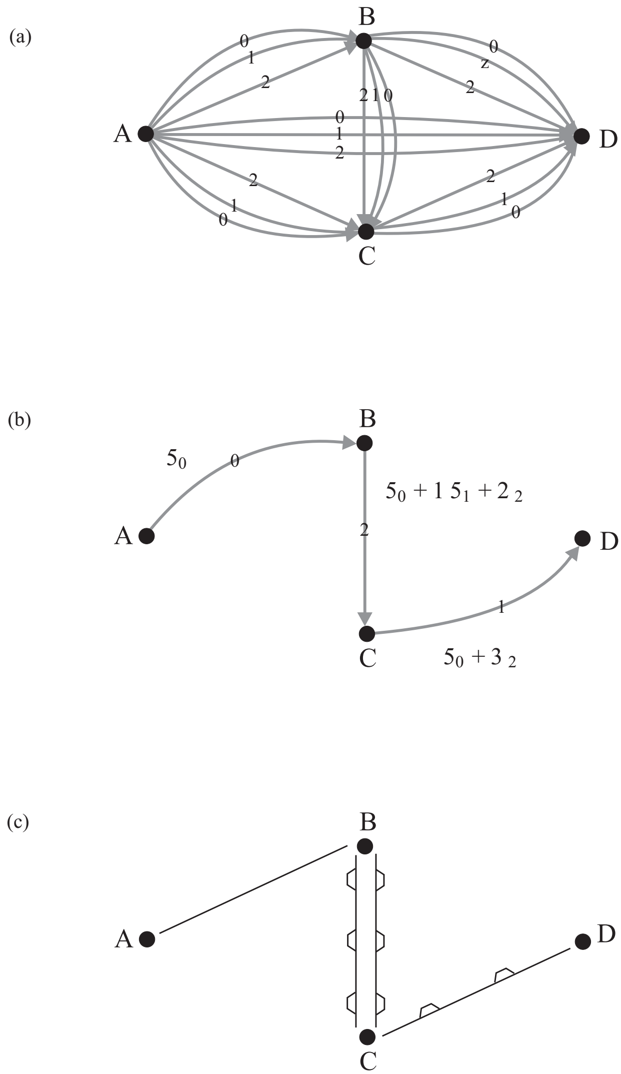

Figure 2: Network design by multi-commodity routing for commodities $C_0 = ((A, D), [5_0, 0_1, 0_2])$, $C_1 = ((B, C), [0_0, 15_1, 2_2])$ and $C_2 = ((C, D), [0_0, 0_1, 3_2])$.

AB_0, BC_2 and CD_1, shown in Figure 2(b). The resulting stages of extension of the lines between the stations are displayed in Figure 2(c).

To calculate the capacity consumption of mixed flows, such as 5_0+3_2, one has to keep in mind that different train types have different characteristics, such as maximum speed, acceleration and deceleration rates, and because of this they consume different amounts of capacity. In terms of flow units this has the impact that, for example, the following holds: $5_i \neq 5_j$. Furthermore, different mix ratios of trains consume different amounts of capacity and there is no train type whose capacity consumption is expressible by a linear combination of the other train types consumptions. This specific characteristic of the capacity consumption is examined in the next section.

2.3 Arc capacity

Capacity consumption of one train on a track section can be expressed using the well-known *minimum headway time*, introduced by Happel [3]. It is the time z_{ij} a train j at least has to wait when it wants to enter a track section that is currently occupied by another train i. A visualization of the minimum headway time using *blocking time stairways* is shown in Figure 3. For more information on the blocking time theory, see Pachl [4].

Wendler [5] calls this minimum headway time in the context of queuing theory *service time* because it is the time frame while one train occupies the service channel – i.e. the track section – and a following train cannot be served.

To calculate the mean minimum headway time of a mix of trains on a track section, the order of trains arriving at the track section is important, since the minimum headway time can only be derived for pairs of consecutive trains. Because SRI deals with future traffic demands there is no fixed timetable for the trains

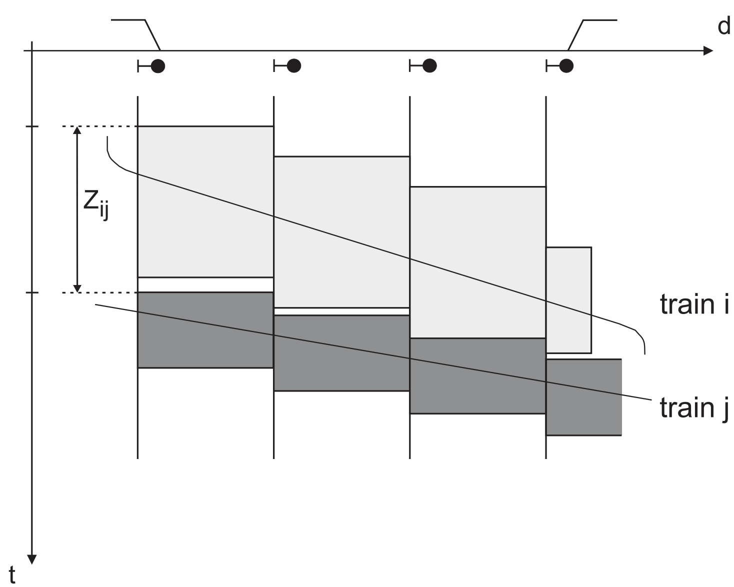

Figure 3: Minimum headway time z_{ij}.

occupying the infrastructure. Because of that, all possible successions ij of trains i and j are considered and weighted with probabilities p_{ij} of the event that train j follows train i. This is shown in eqn (9), where x_i is the number of trains of type i, x_j is the number of trains of type j, and N is the number of all trains. This is a very simple estimation, which will be refined in future SRI solver implementations. The mean minimum headway time $\bar{z}_{ij}$ and the expected service time ET_B over all trains in the time horizon, respectively, can be derived by summing up products $p_{ij} \cdot x_{ij}$ (eqn (10)). Eqn (11) shows the expected capacity consumption on a given track section a and mix of trains of different train types $|K|$.

$$p_{ij} := \frac{x_i}{N} \cdot \frac{x_j}{N} = \frac{x_i \cdot x_j}{N^2}, \tag{9}$$

$$\bar{z}_{ij} = ET_B = \sum_i \sum_j p_{ij} \cdot z_{ij}, \tag{10}$$

$$N^a \cdot ET_B^a = N^a \cdot \sum_{i \in K} \sum_{j \in K} \frac{x_i^a \cdot x_j^a}{{N^a}^2} \cdot z_{ij}^a \tag{11}$$

By means of the preceding definitions it is now possible to state the SRI problem as an optimization problem.

2.4 SRI: multi-commodity flow with multi-arcs

For a network $G = (N, A)$, a set of commodities C and set of flow types (train types) FT the optimization model of the SRI is formulated as follows:

$$\text{Minimize} \sum_{a \in A} c^a \cdot x_{used}^a \tag{12}$$

subject to

$$N^a \cdot \sum_{i \in K} \sum_{j \in K} \frac{x_i^a \cdot x_j^a}{{N^a}^2} \cdot z_{ij}^a \leqslant 0.6 \cdot t_U \qquad \forall a \in A, \tag{13}$$

$$\sum_{a \in Out\,(n)} x_{ij}^a - \sum_{a \in In\,(n)} x_{ij}^a = b_{ij}^n \qquad \forall n \in N, \forall i \in FT, \forall j \in C \tag{14}$$

$$x_{ij}^a \geqslant 0 \qquad \forall a \in A, \forall i \in FT, \forall j \in C \tag{15}$$

$$x_{\text{used}}^a = \begin{cases} 1, & \text{if } \exists i : x_a^i > 0, \\ 0, & \text{else.} \end{cases} \tag{16}$$

where x_i^a is the sum of all flows on arc a of the flow type for train type i, and x_{ij}^a is the flow of type i of commodity j on arc a. For a node n, the functions In and Out return the set of all arcs $(n', n) \in A$ and $(n', n) \in A$, respectively.

Because one unit of flow corresponds to one train, it seems to be incorrect to not restrict the variable domain to positive integers. However, since the used traffic demands are derived from future traffic flow estimations it is sufficient to choose positive reals as the domain, as stated in eqn (15). This relaxation furthermore simplifies the solving process. The objective is to minimize the life cycle cost for the arcs that are used to route flow. The variables x^a_{used}, defined in eqn (16), are used to ensure that arc costs arise if and only if there is flow on the arc. Usually, cost functions in flow problems depend on the amount of flow, but since the track, which corresponds to an arc, has to be constructed independently of the number of trains running on it, the cost model described above is chosen. Eqn (14) contains the known flow conservation constraint, cf. eqns (2) and (7). Eqn (13) is called capacity constraint. The left-hand side describes the occupation time of the mix of trains routed via track/arc a, cf. eqn (11). This capacity consumption is bounded to an amount of 60% of the observed time frame t_U. This value is taken out of UIC Code 406 [6]. This is a leaflet of the International Union of Railways, which standardizes railway capacity analysis. There exist of course more sophisticated capacity models, but for the sake of simplicity the approach according to UIC Code 406 is selected. Since the capacity constraint is non-linear, powerful LP/MIP-solvers are not applicable. To overcome this difficulty the model is transformed into a mixed integer program.

2.5 SRI: MIP model

A MIP is a linear optimization problem that contains variables with an integer value domain. Because of this integrality a MIP is much more difficult to solve than a LP. The MIP resulting from the following transformation of the optimization problem given in eqns (12)–(16) is furthermore a binary MIP (BMIP), because the integral variables are even binary variables.

To get rid of the non-linear capacity constraint, possible train/flow type mix ratios that fully utilize the capacity of an arc are calculated for each arc, using ET_B, see eqn (11). These mix ratios are denoted as configurations. In the case of three different train types 0, 1 and 2, the set of configurations for an exemplary arc a has the following form:

$$Conf^a := \{[76_0, 0_1, 0_2], [74_0, 1_1, 0_2], ..., [0_0, 64_1, 0_0], ..., [0_0, 0_1, 42_2]\}. \quad (17)$$

In the MIP model there is one binary variable y^a_c for each configuration $c \in Conf_a$. This holds for every arc $a \in A$. For each arc a at most, one of these variables can be selected by setting its value to 1. This enables the chosen configuration and for each flow type i the sum of the flow of type i on that arc is bounded by the value of the ith component of the selected configuration. This means that one configuration $[x_0, x_1, x_2]^a$, if selected for an arc a, restricts the flow on a for each flow type i to the value of x_i. This is modeled by the new linear capacity constraint in eqn (19) of the MIP model. The objective is to minimize the cost of the selected arcs, which is equal to maximizing the cost of the arcs that are not picked. Therefore, variables $y^a_{\text{off/on}}$ are introduced and set to 1 if the arc is not used.

This restructuring of the objective function provides a faster solving process. The constraint shown in eqn (21) ensures that an arc is either switched off or exactly one configuration is enabled.

$$\text{Maximize} \sum_{a \in A} c^a \cdot y^a_{\text{off/on}} \tag{18}$$

subject to

$$\sum_{j \in C} x^a_{ij} \leqslant \sum_{c \in Conf^a} y^a_c \cdot v^c(i) \qquad \forall a \in A, \forall i \in FT, \tag{19}$$

$$\sum_{a \in \text{Out}\,(n)} x^a_{ij} - \sum_{a \in \text{In}\,(n)} x^a_{ij} = b^n_{ij} \qquad \forall n \in N, \forall i \in FT, \forall j \in C \tag{20}$$

$$\sum_{c \in Conf^a} y^a_c + y^a_{\text{off/on}} = 1 \qquad \forall a \in A \tag{21}$$

$$x^a_{ij} \geqslant 0 \qquad \forall a \in A, \forall i \in FT, \forall j \in C \tag{22}$$

$$y^a_c \in \{0,1\} \qquad \forall a \in A, \forall c \in Conf^a \tag{23}$$

$$y^a_{\text{off/on}} \in \{0,1\} \qquad \forall a \in A, \tag{24}$$

where $G = (N, A)$ is the network, $Conf^a$ the set of all configurations associated with arc a, C the set of commodities, FT the set of flow types and function v^c returns for a given $i \in FT$ the value of the ith component of the configuration c.

3 Solving

The previous section presents a MIP model of the SRI problem. To solve the problem professional solver software, which provides academic user licenses, is used. Two solvers with different advantages were used to solve the problem as follows.

3.0.1 Gurobi

The Gurobi Optimizer is a linear programming mixed integer programming solver that exploits modern multi-core processors. Gurobi is currently the performance benchmark winner, so it provides the fastest solving times. The disadvantage of Gurobi is the interface. It allows only the usage of restricted sets of functions, parameters and attributes, which can be accessed via the programming languages C, C++, Java, .NET or Python. Despite this restriction it is a powerful solver that additionally supports some modeling systems, such as MPL and AMPL, and is able to read and write LP and MPS files. For further information see the Gurobi homepage [7].

3.0.2 SCIP

SCIP stands for Solving Constraint Integer Programs and was developed at the Konrad-Zuse-Zentrum for information technology in Berlin. Since the complete source code is available the solver allows total control of the solution process and unrestricted access to any information at any stage of the solution process. The user can define, write and include their own pricers, branching rules, presolvers, heuristics and so on. Basic principles and further information about the concept of constraint integer programming and SCIP are provided by Achterberg [8] and the SCIP homepage [9].

3.1 Solving mixed integer problems

There is a wide range of methods that are useful for solving (mixed) integer programs efficiently. The kernel method, which is state of the art and the cause of that focussed here, is called branch-and-bound.

3.1.1 Branch-and-bound

The goal of the branch-and-bound method is to find an assignment of values of the integer variables that forms an optimal solution of the MIP. One way to achieve this is to enumerate all possible assignments of values by a so called explicit enumeration tree. This results, even in the binary case for a small set variables, in a huge number of tree nodes. So it is desirable not to explore the whole tree. To achieve a so called implicit enumeration tree, bounds are calculated at each node of the branch-and-bound tree during tree building. With the help of these bounds it is possible to prune branches of the tree, so that they need not be explored. The most common method for finding bounds is to solve the linear programming relaxation of the given MIP. In the case of a maximization problem the optimal solution of the relaxation provides an upper bound on every solution of the MIP and is the basis for the branching decision in the current node. Branching in the binary case means creating two new branches of the branch-and-bound tree by assigning the values 0 and 1 to the variable that is chosen to branch on. For more detailed information about branch-and-bound, see Wolsey [10].

3.2 Results

Current implementations run on examples with 20 stations, 3 different train types, 4 different track types and 10 traffic flows. Gurobi returns a result within a 1% optimality gap in less than 7 minutes. The optimality gap is calculated using the best upper in best lower bound in the current stage of the solving process. Proving the optimality, i.e. reaching a gap of 0%, currently takes a great deal of time. This is caused by the huge amount of binary variables. The described example contains about $2 \cdot 10^6$ binary variables and $4.5 \cdot 10^4$ continuous variables. To overcome this explosion in the number of binary variables ongoing implementations are focussed on approaches such as column generation, described by Desrosiers and Lübbecke [11], which try to minimize the number of binary variables needed to calculate the optimal solution.

4 Conclusions

This paper shows how to synthesize networks of railway infrastructure out of estimated future traffic flows. To do so the problem is modeled as an optimization problem by interpreting it as a multi-commodity flow problem on a complete graph with multi-arcs, so that a found routing determines the arcs needed. To make the optimization model applicable to professional solver software the problem is transformed to a MIP and respectively BMIP. This transformation results in a large number of binary variables, which again results in long solver running times. To overcome this difficulty ongoing research focusses on approaches, such as column generation, which try to minimize the number of binary variables needed to calculate the optimal solution.

Acknowledgement

This publication is a result of research done in the DFG Research Training Group 1298, AlgoSyn: *Algorithmic synthesis of reactive and discrete-continuous systems*, funded by the Deutsche Forschungsgemeinschaft (DFG).

References

[1] Ross, S., *Strategische Infrastrukturplanung im Schienenverkehr*. Deutscher Universitäts-Verlag: Wiesbaden, 2001.
[2] Ahuja, R.K., Magnanti, T.L. & Orlin, J.B., *Network flows : Theory, Algorithms, and Applications*. Prentice Hall: Upper Saddle River, New Jersey, pp. 649–686, 1993.
[3] Happel, O., Sperrzeiten als Grundlage für die Fahrplankonstruktion. *ETR*, **8(2)**, pp. 79–90, 1959.
[4] Pachl, J., Timetable design principles (Chapter 2). *Railway Timetable & Traffic*, eds. I.A. Hansen & J. Pachl, Eurailpress: Hamburg, pp. 9–42, 2008.
[5] Wendler, E., Timetable design principles (Chapter 6). *Railway Timetable & Traffic*, eds. I.A. Hansen & J. Pachl, Eurailpress: Hamburg, pp. 106–117, 2008.
[6] International Union of Railways: UIC Code 406 - Capacity, pp. 18–20, 2004.
[7] http://www.gurobi.com/, May 2010.
[8] Achterberg, T., *Constraint Integer Programming*. Ph.D. thesis, Technische Universität Berlin, Berlin, 2007.
[9] http://scip.zib.de/, May 2010.
[10] Wolsey, L.A., *Integer Programming*. Wiley-Interscience: New York, pp. 91–108, 1998.
[11] Desrosiers, J. & Lübbecke, M.E., A primer in columnn generation (Chapter 1). *Column Generation*, eds. G. Desaulniers, J. Desrosiers & M.M. Solomon, Springer: New York, pp. 1–32, 2005.